Distributed File Systems

ECE 454 / 751: Distributed Computing

Instructor: Dr. Wojciech Golab

wgolab@uwaterloo.ca

Slides are derived mostly from A. S. Tanenbaum and M. Van Steen,
Distributed Systems: Principles and Paradigms, 2nd Edition, Pearson-Prentice Hall, 2006.

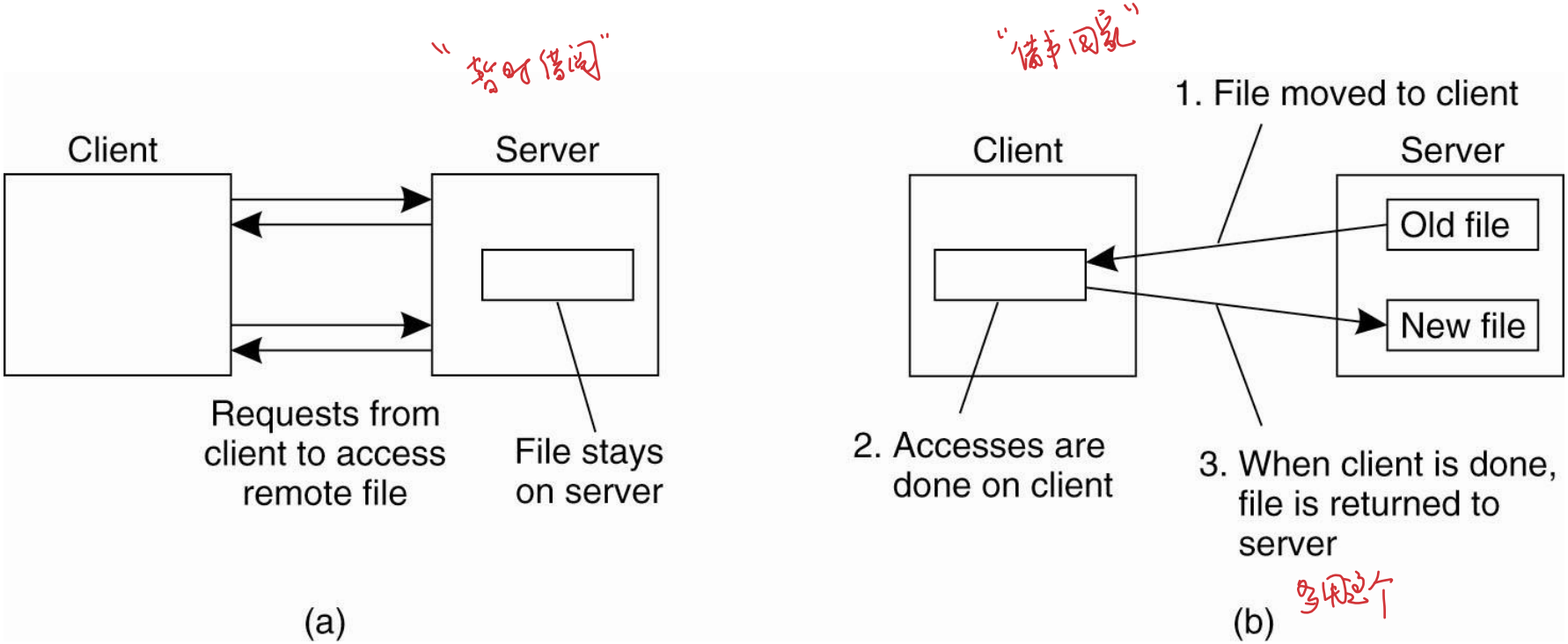
Learning objectives

To develop a conceptual understanding of:

- client-server interactions in distributed file systems
- NFS network file system
- GFS / HDFS
- file sharing semantics

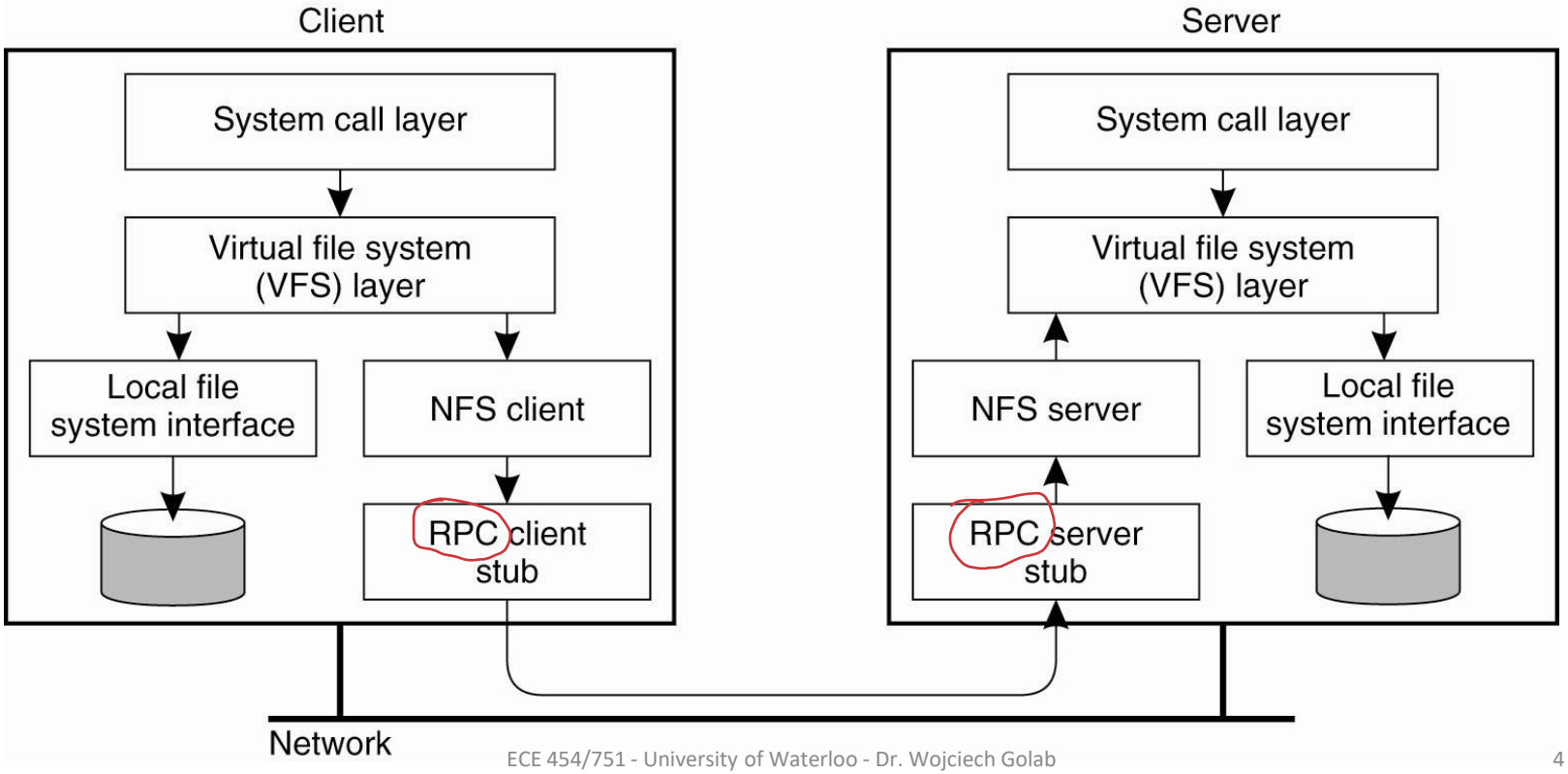
Accessing remote files

Client-server interactions in a distributed file system (DFS) may follow the **remote access model** (a), or the **upload/download model** (b).



Network File System (NFS)

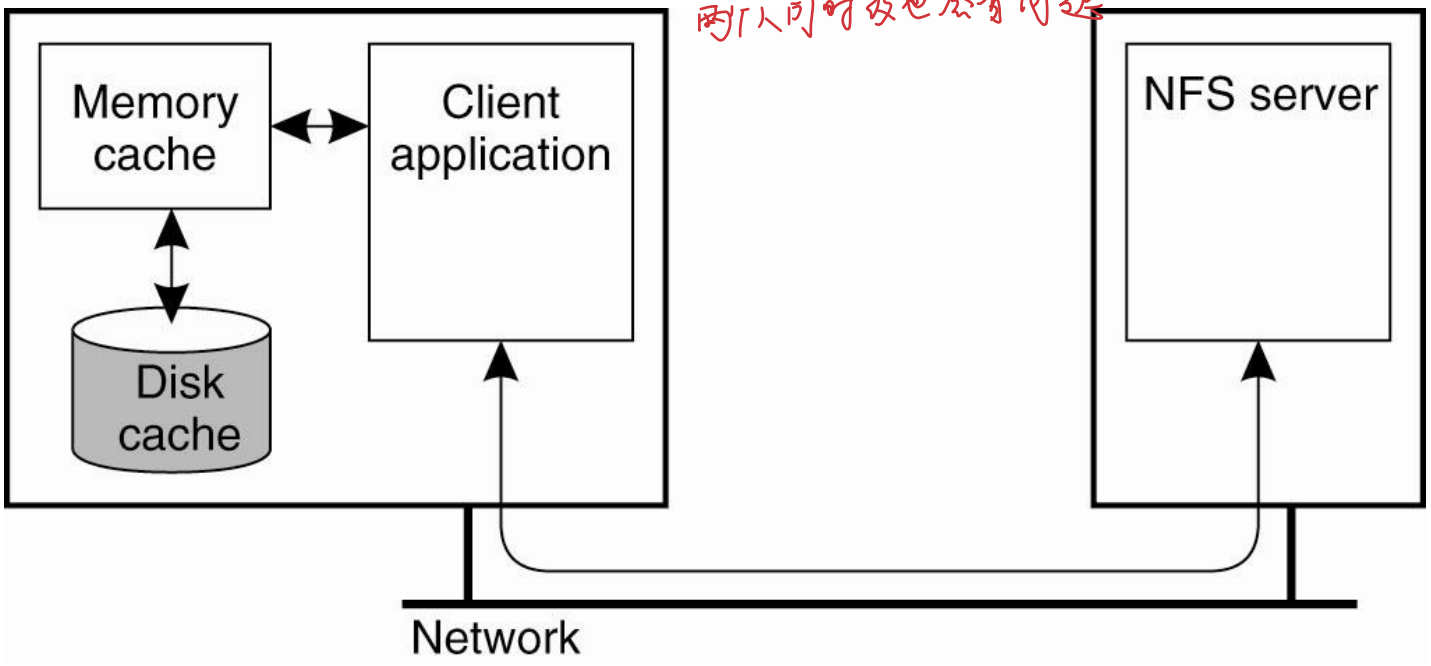
Unix-like systems rely heavily on NFS, which was originally developed at Sun Microsystems in 1984. For example, your home directory on ecelinux is mounted remotely using NFSv4.



Network File System (NFS)

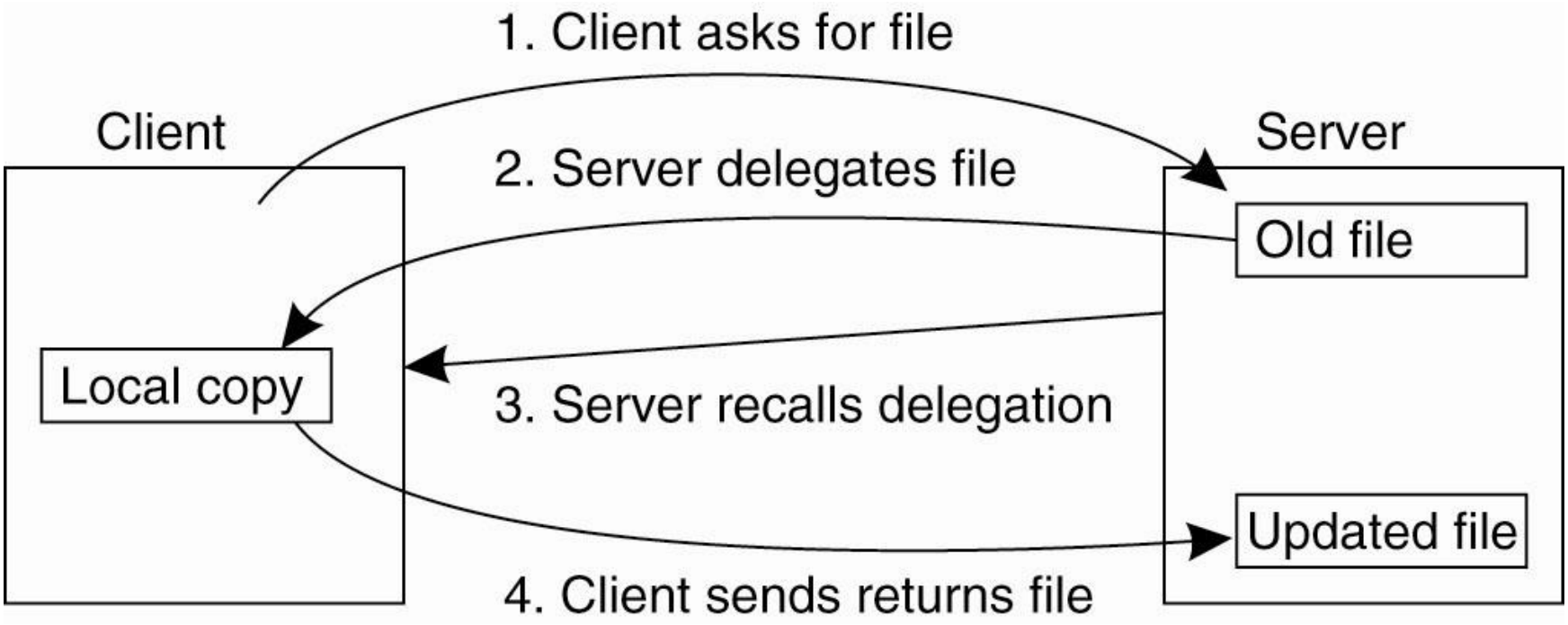
NFS supports client-side caching to reduce communication between client and server. Modifications are flushed back to the server when the client closes the file. Consistency is handled in an implementation-dependent way.

关上文件才会修改. 存在缓存中.
两人同时改也会有问题



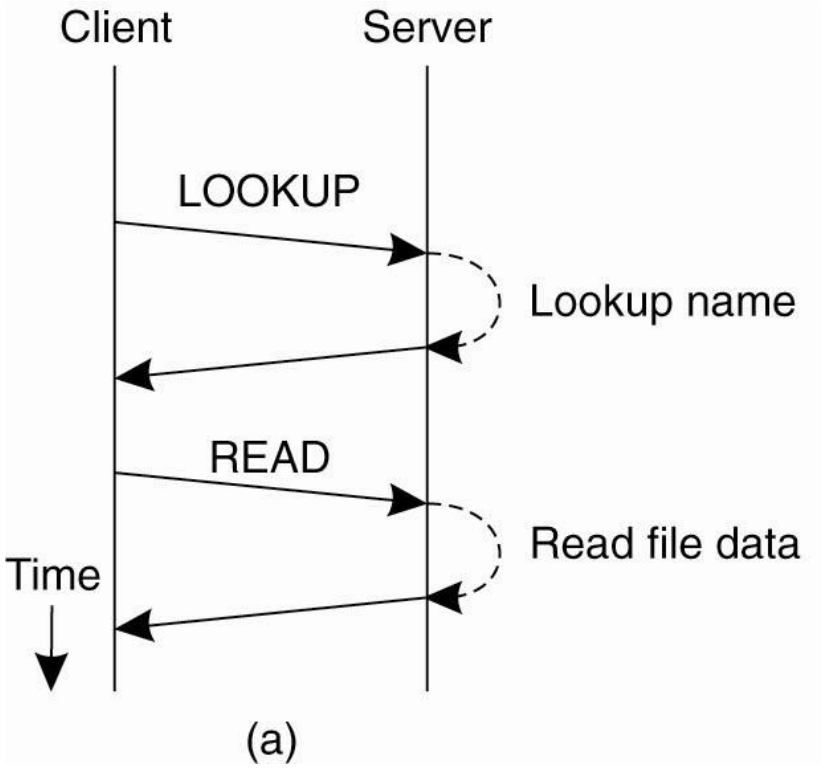
Network File System (NFS)

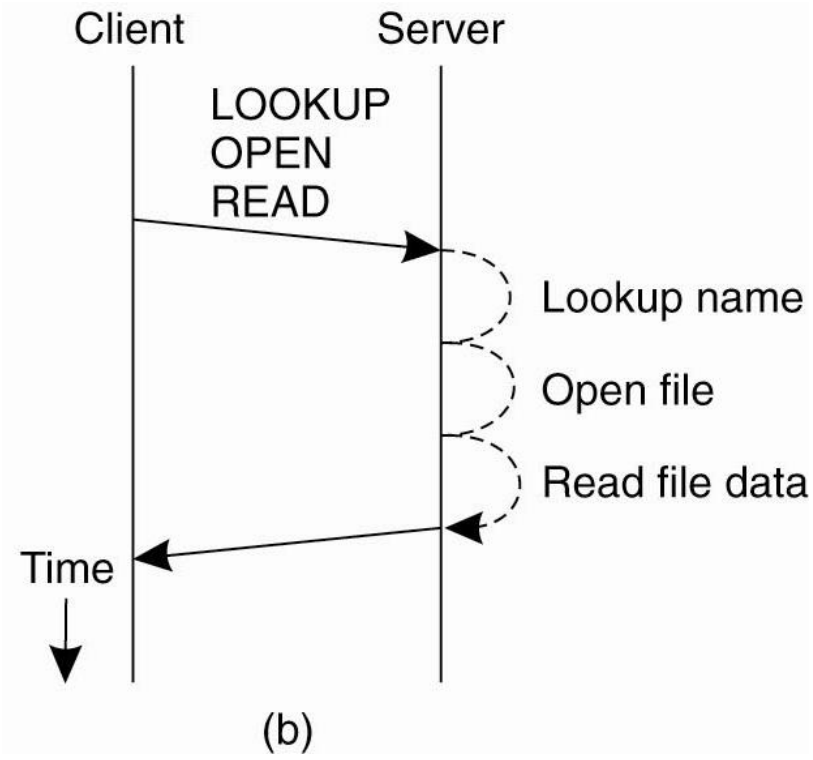
NFSv4 further allows servers to delegate authority over a file to clients, and provides a callback mechanism to recall delegation.



Network File System (NFS)

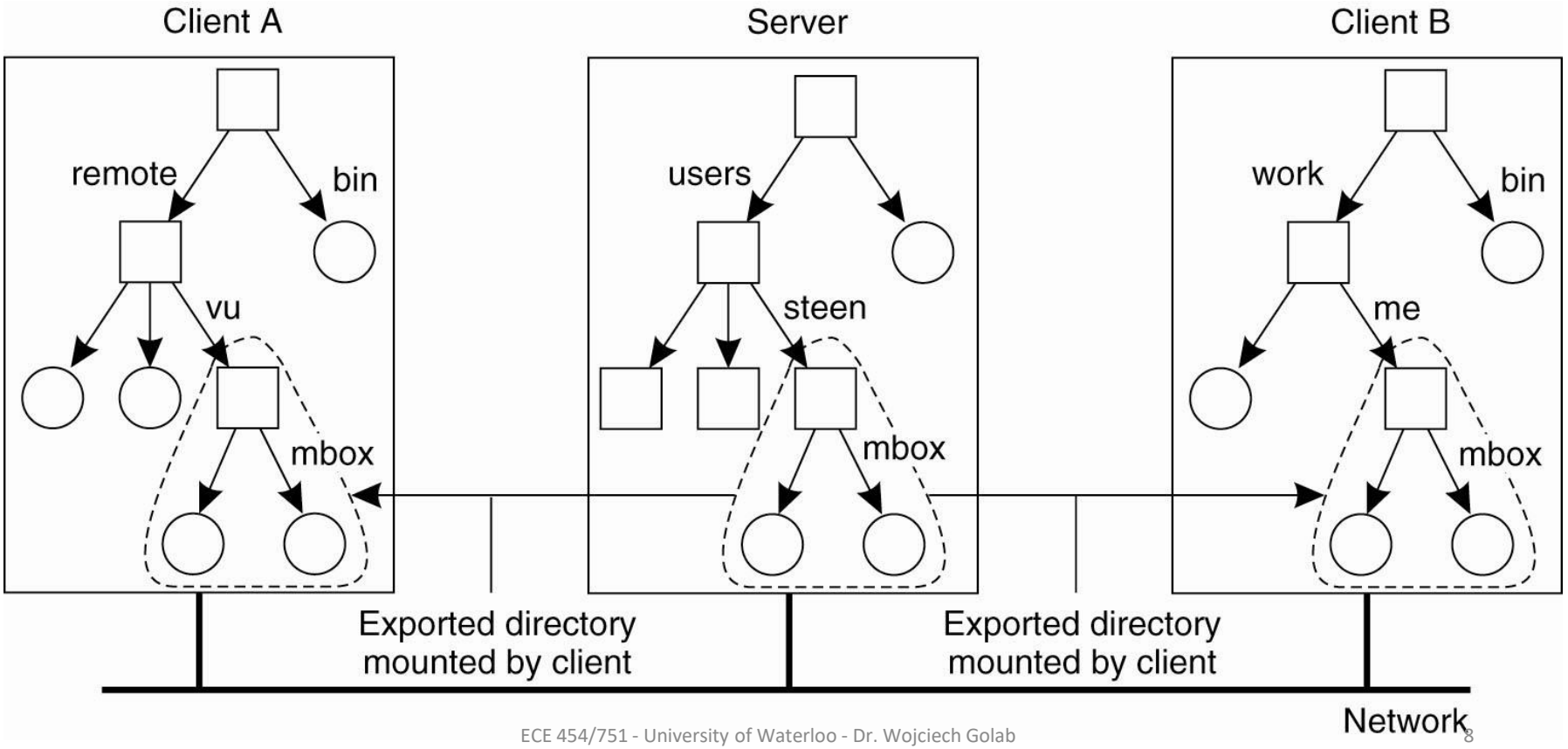
NFS uses RPCs internally. File system operations in v3 execute in multiple round trips (a). NFSv4 supports **compound procedures** (b).





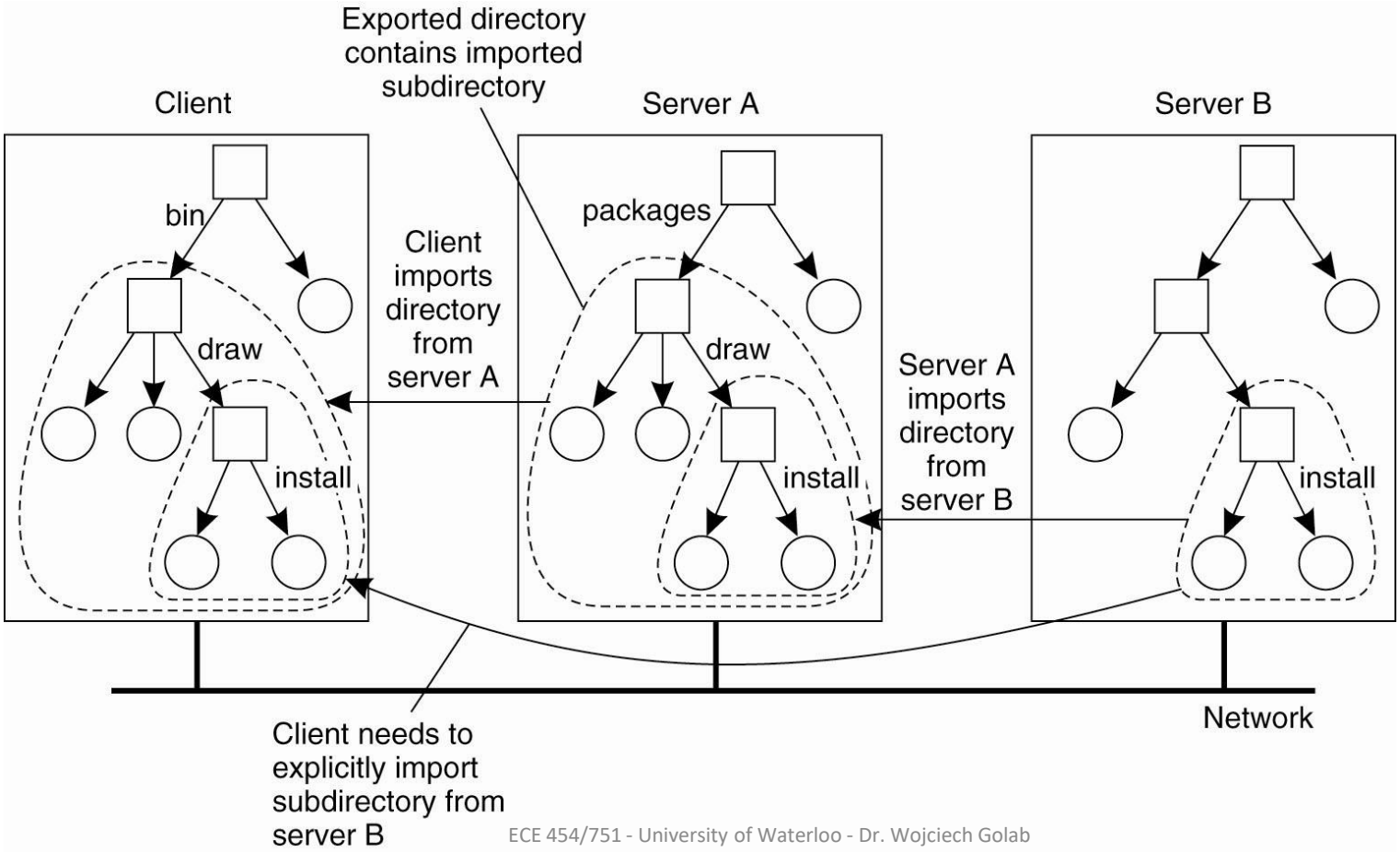
Network File System (NFS)

An NFS server generally exports only a part of its local file system to remote clients.



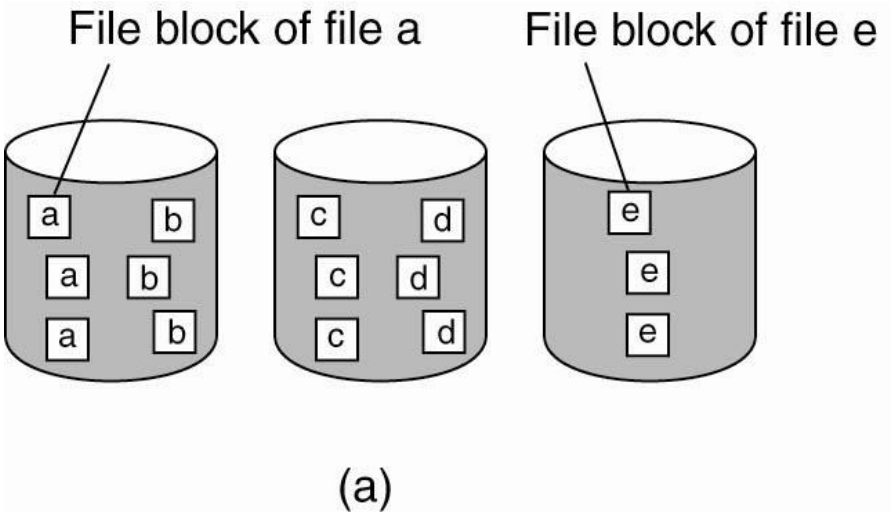
Network File System (NFS)

A client may access nested directories exported by multiple servers.

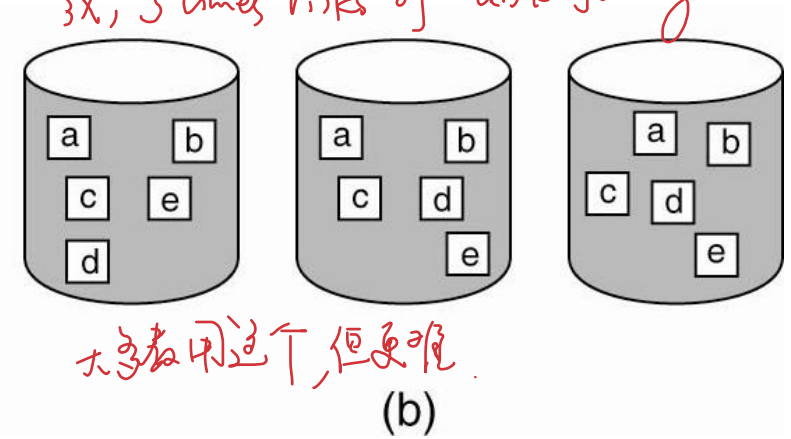


Distributing large files

In typical NFS deployments, one NFS server shares files with many client hosts. A large scale DFS may instead distribute files across multiple servers. Each file may reside at one server (a), or may be striped across servers (b) somewhat similarly to striping in RAID.



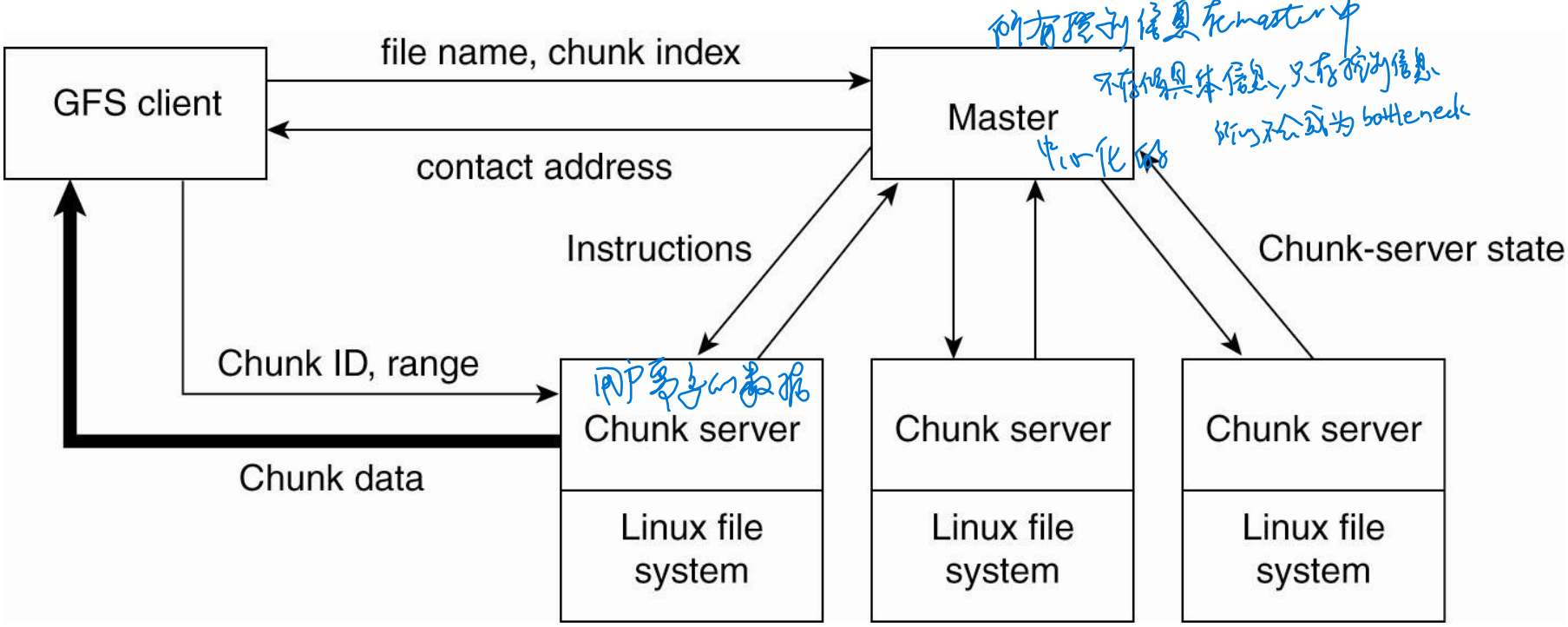
(a)



(b)

Google File System (GFS)

GFS is a DFS that stripes files across inexpensive commodity servers without RAID. GFS is layered on top of ordinary Linux file systems, and provides fault tolerance in software. The Hadoop Distributed File System (HDFS) is a simplified open-source implementation of GFS.



Google File System (GFS)

https://lianhaimiao.github.io/2018/03/10/%E7%90%86%E8%A7%A3Google-File-System/

- The GFS master stores meta-data about files and chunks, and serves it to clients. The meta-data is cached in main memory and updates are logged to local storage. Checkpoints reduce the length of the log.
- The GFS master periodically polls the chunk servers to keep the meta-data consistent. Some inconsistencies are possible but clients are generally able to locate at least one copy of a given chunk thanks to replication.
- To **read** data from a file, a client sends the file name and chunk index to the master, who responds with a contact address. The client then pulls data directly from a chunk server, bypassing the master, as shown in the last slide.

Google File System (GFS)

To **update** data in a file, a client contacts the nearest chunk server holding the data, and pushes its updates to that server. This server will push the update to the next closest one holding the data, and so on, in a pipelined fashion. Once all replicas have received the data, the primary chunk server assigns a sequence number to the update operation and passes it on to the secondary chunk servers. The master is once again bypassed. Finally, the primary replica informs the client that the update is complete.

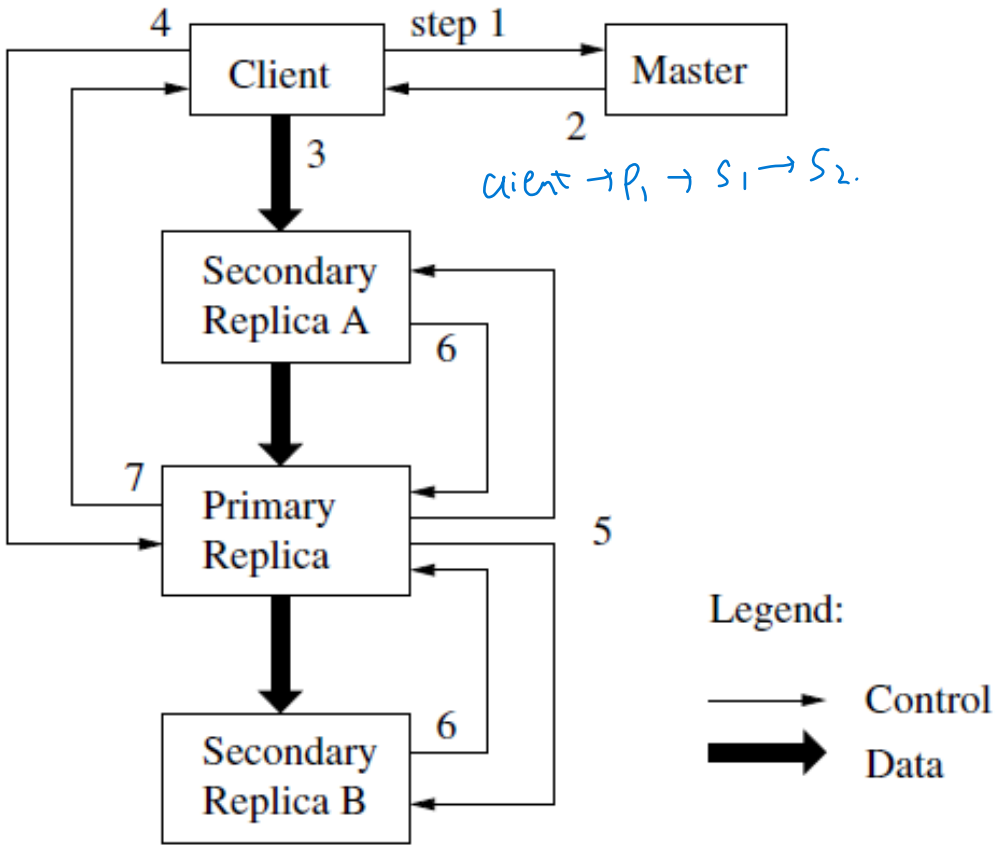


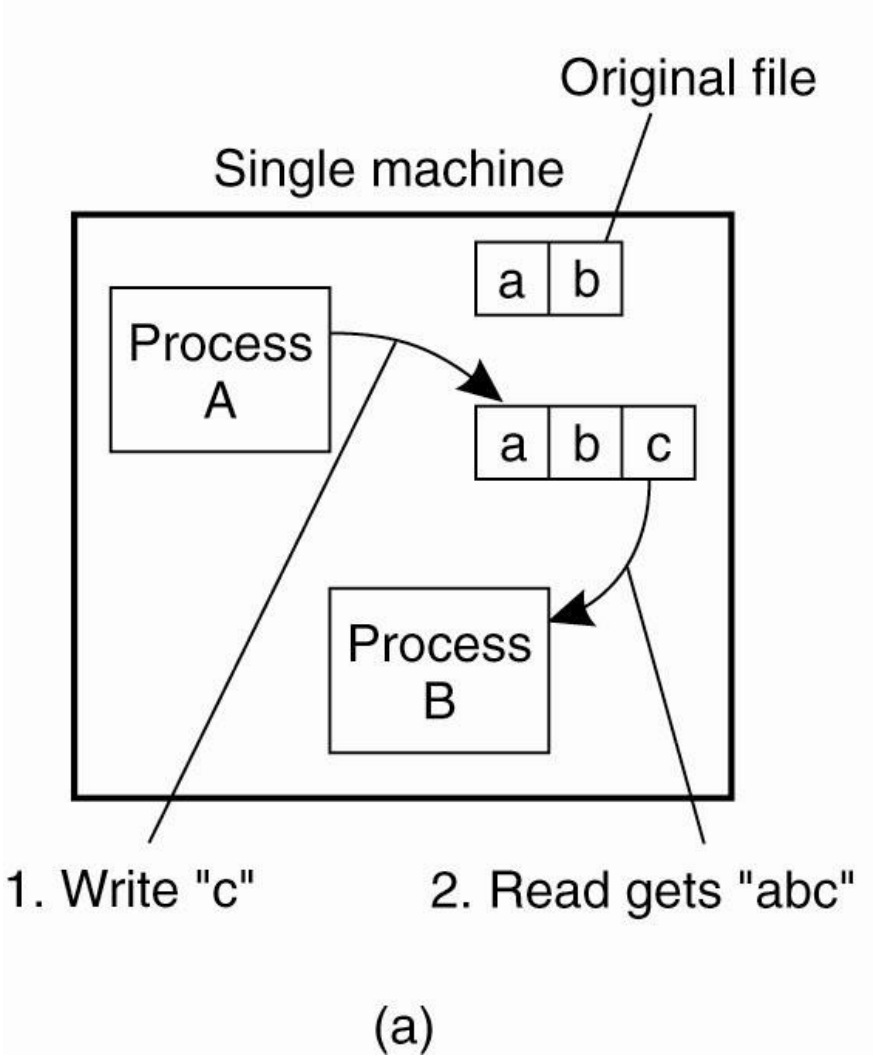
Figure 2: Write Control and Data Flow

source: Ghemawat et al., SOSP 2003

Semantics of file sharing

In a centralized setting, the semantics of file sharing usually state that file system operations are strictly ordered in time. This ensures (among other things) that an application can always read its own writes.

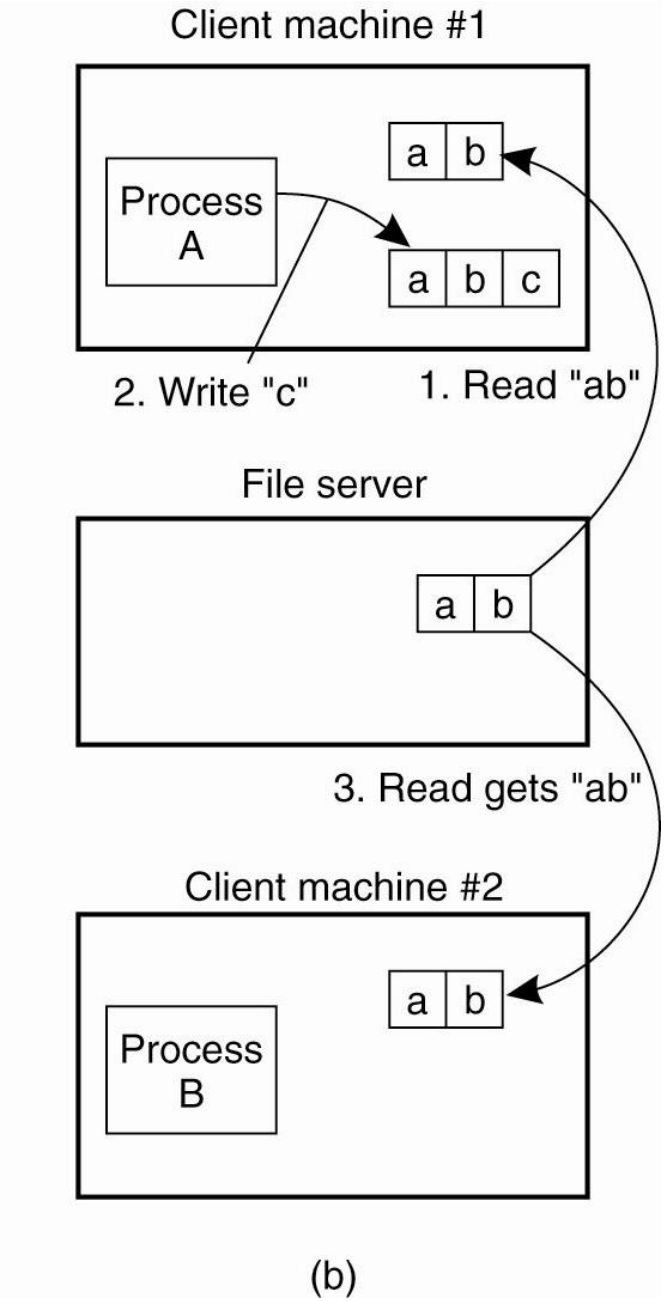
The same behavior, called **UNIX semantics**, can be achieved in a DFS as long as there is only one file server and files are not cached.



Semantics of file sharing

When a cached file is modified in a DFS, it is possible but impractical to propagate the changes immediately to the file server.

Alternatively, the modifications can be made visible only to the process or machine that modified the file. The modifications are made visible to other processes when the file is closed, and the final version of the file is determined by the last client that closes the file. This rule is widely implemented and known as **session semantics**.



Semantics of file sharing

语义

The semantics of file sharing in a DFS can be defined in more than one way. NFSv4 supports session semantics + byte range file locking. Locks in NFS have an associated lease that simplifies recovery after failures. HDFS instead provides immutable files, but supports an append function so that the file system can be used effectively for storing log-structured data.

Method	Comment
UNIX semantics	Every operation on a file is instantly visible to all processes
Session semantics	No changes are visible to other processes until the file is closed
Immutable files 不可更改的	No updates are possible; simplifies sharing and replication
Transactions	All changes occur atomically

lock entire / a part