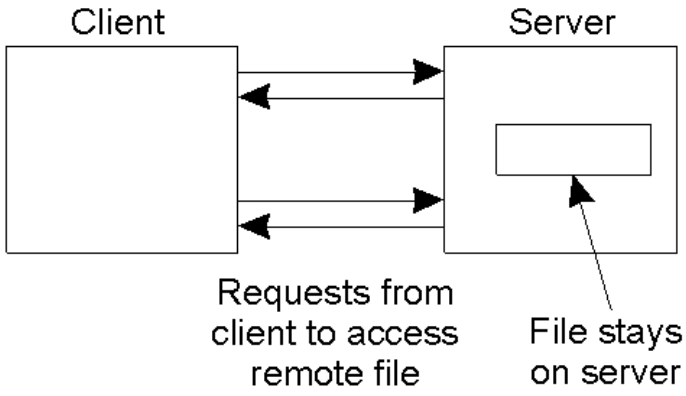
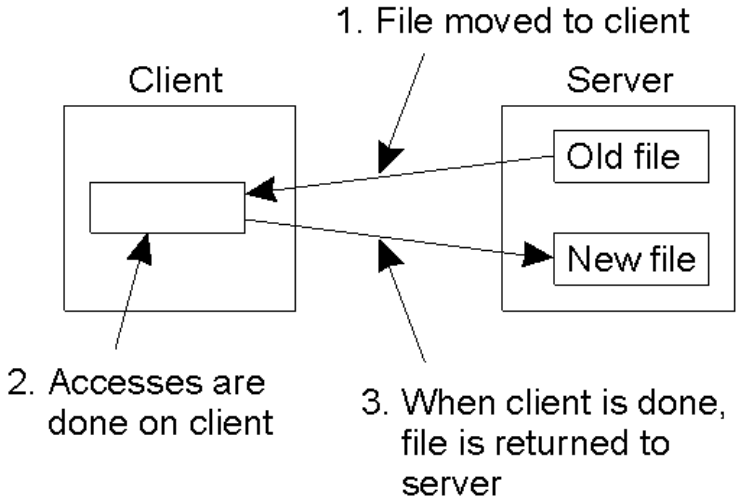
Distributed File Systems

NFS Architecture (1)

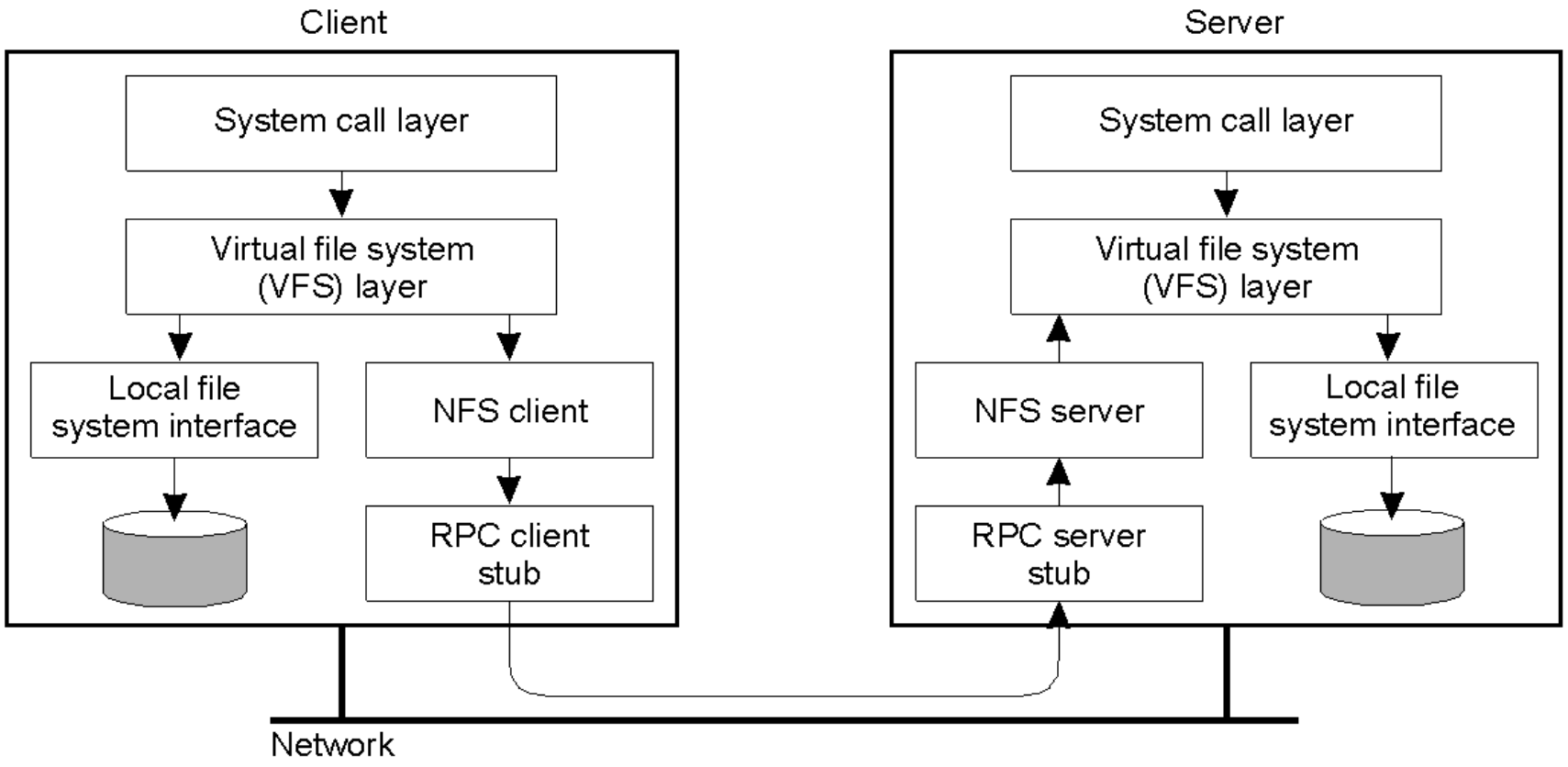




a) The remote access model.

b) The upload/download model

NFS Architecture (2)



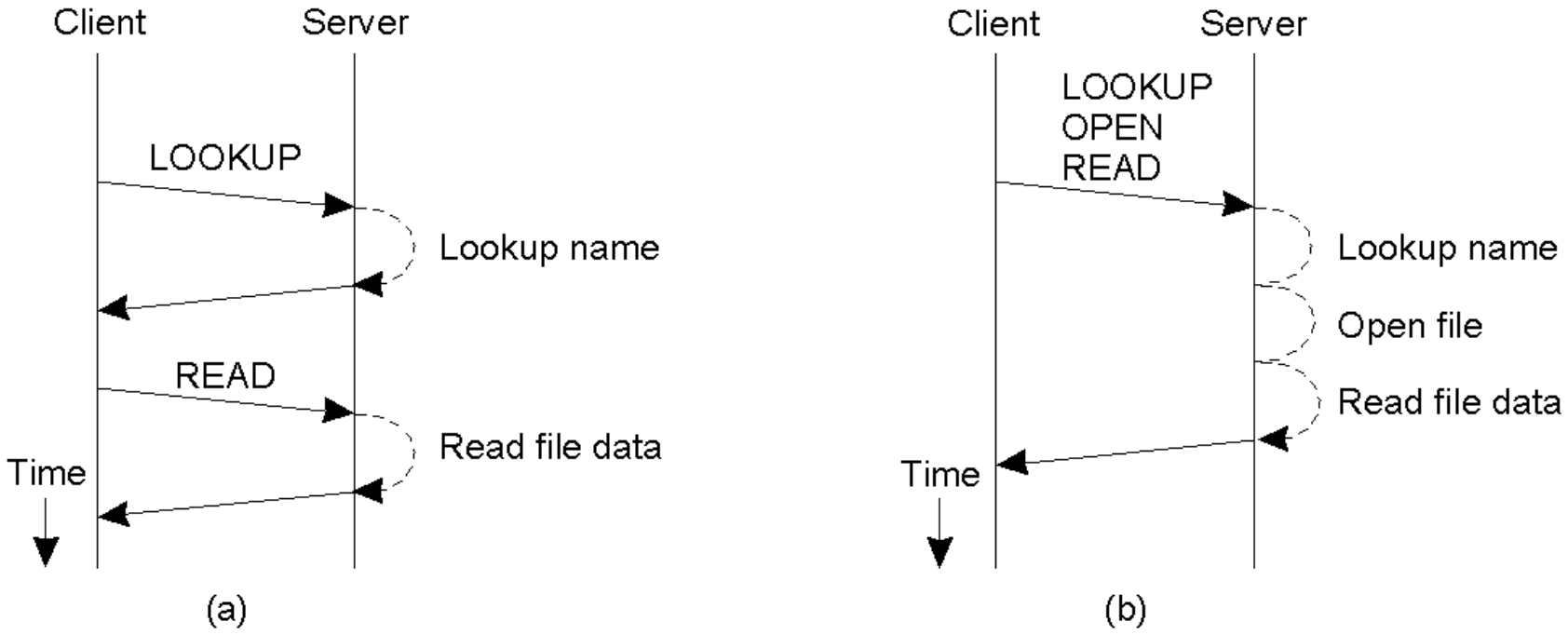
The basic NFS architecture for UNIX systems.

File System Model

Operation	v3	v4	Description
Create	Yes	No	Create a regular file
Create	No	Yes	Create a nonregular file
Link	Yes	Yes	Create a hard link to a file
Symlink	Yes	No	Create a symbolic link to a file
Mkdir	Yes	No	Create a subdirectory in a given directory
Mknod	Yes	No	Create a special file
Rename	Yes	Yes	Change the name of a file
Rmdir	Yes	No	Remove an empty subdirectory from a directory
Open	No	Yes	Open a file
Close	No	Yes	Close a file
Lookup	Yes	Yes	Look up a file by means of a file name
Readdir	Yes	Yes	Read the entries in a directory
Readlink	Yes	Yes	Read the path name stored in a symbolic link
Getattr	Yes	Yes	Read the attribute values for a file
Setattr	Yes	Yes	Set one or more attribute values for a file
Read	Yes	Yes	Read the data contained in a file
Write	Yes	Yes	Write data to a file

An incomplete list of file system operations supported by NFS.

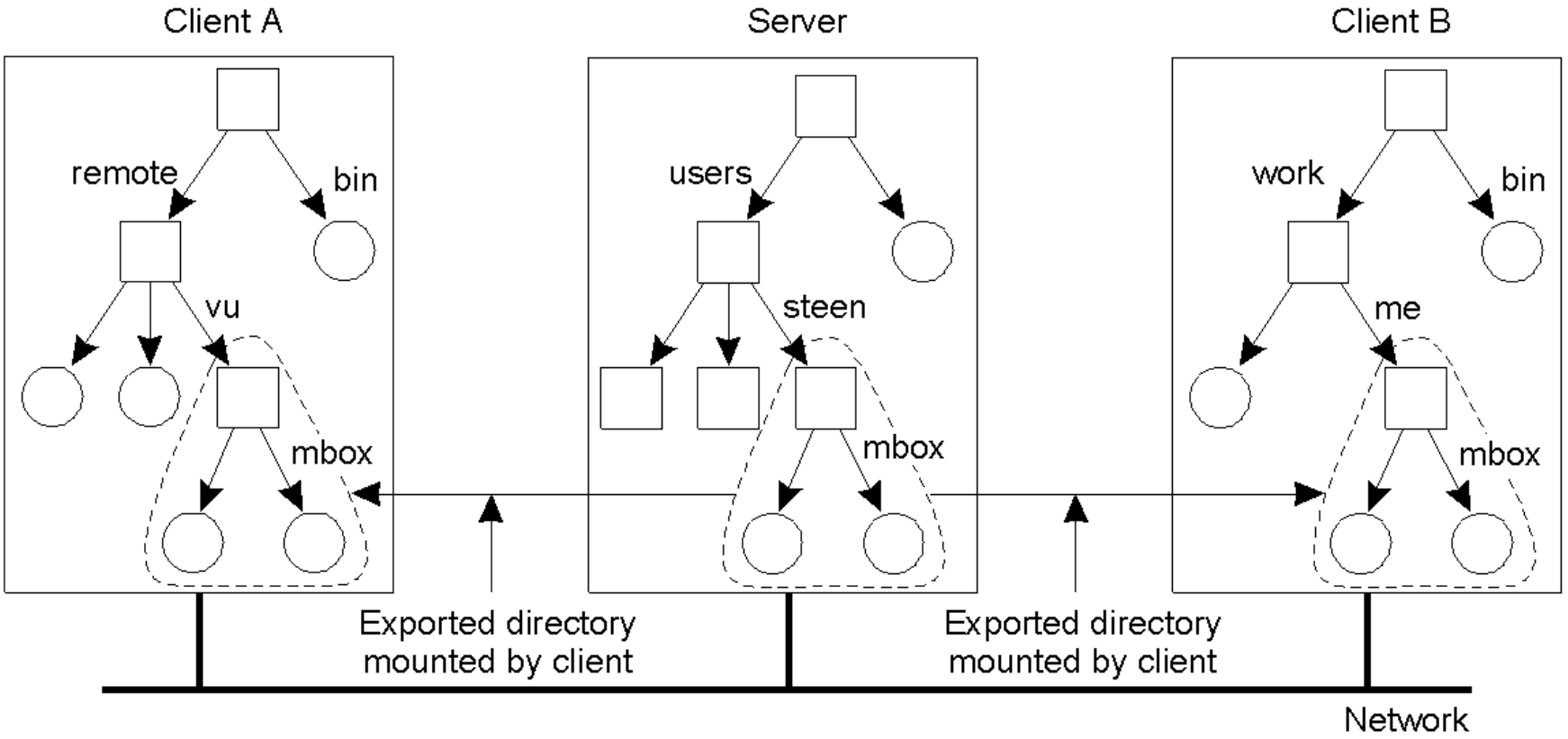
Communication



a) Reading data from a file in NFS version 3.

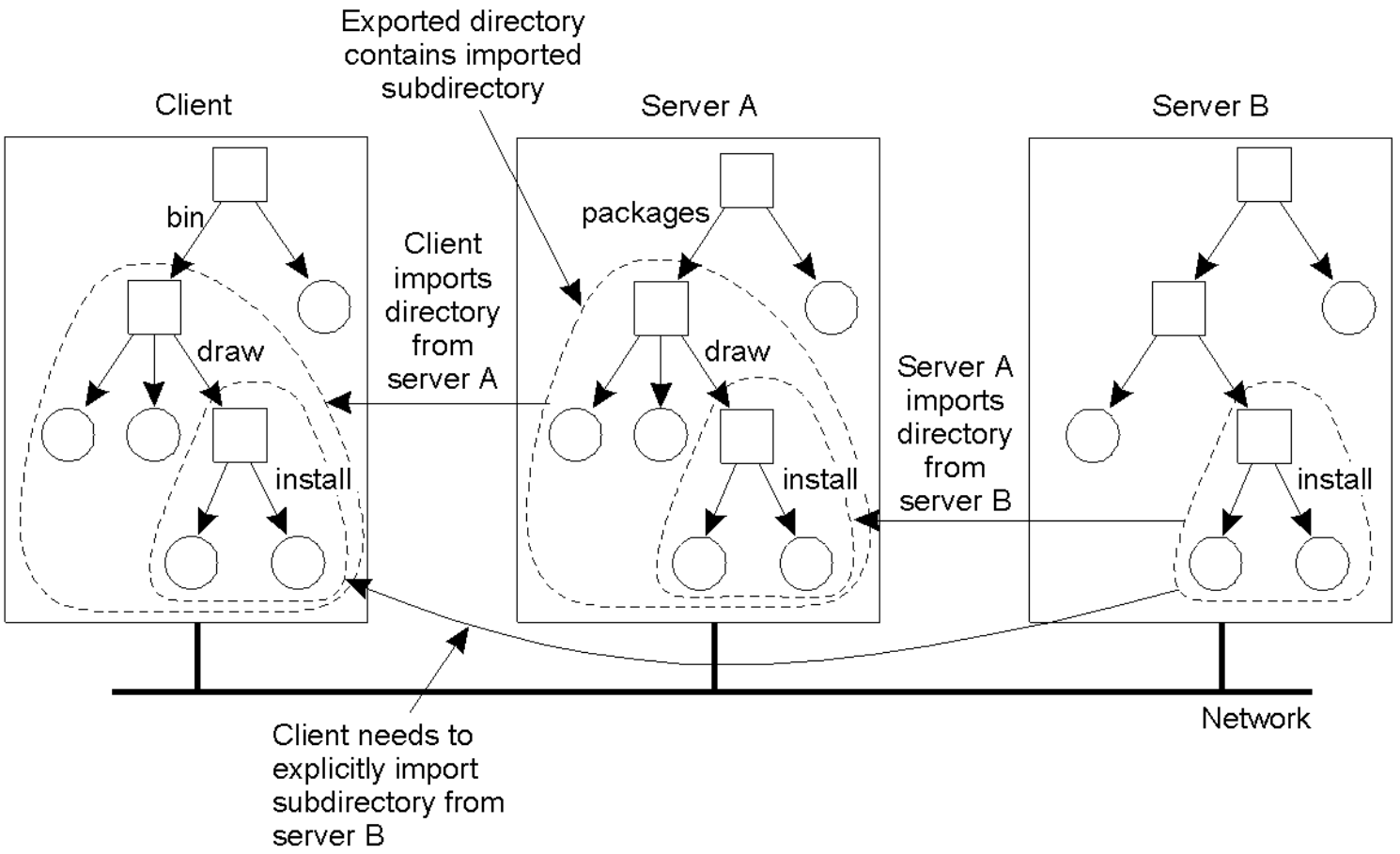
b) Reading data using a compound procedure in version 4.

Naming (1)



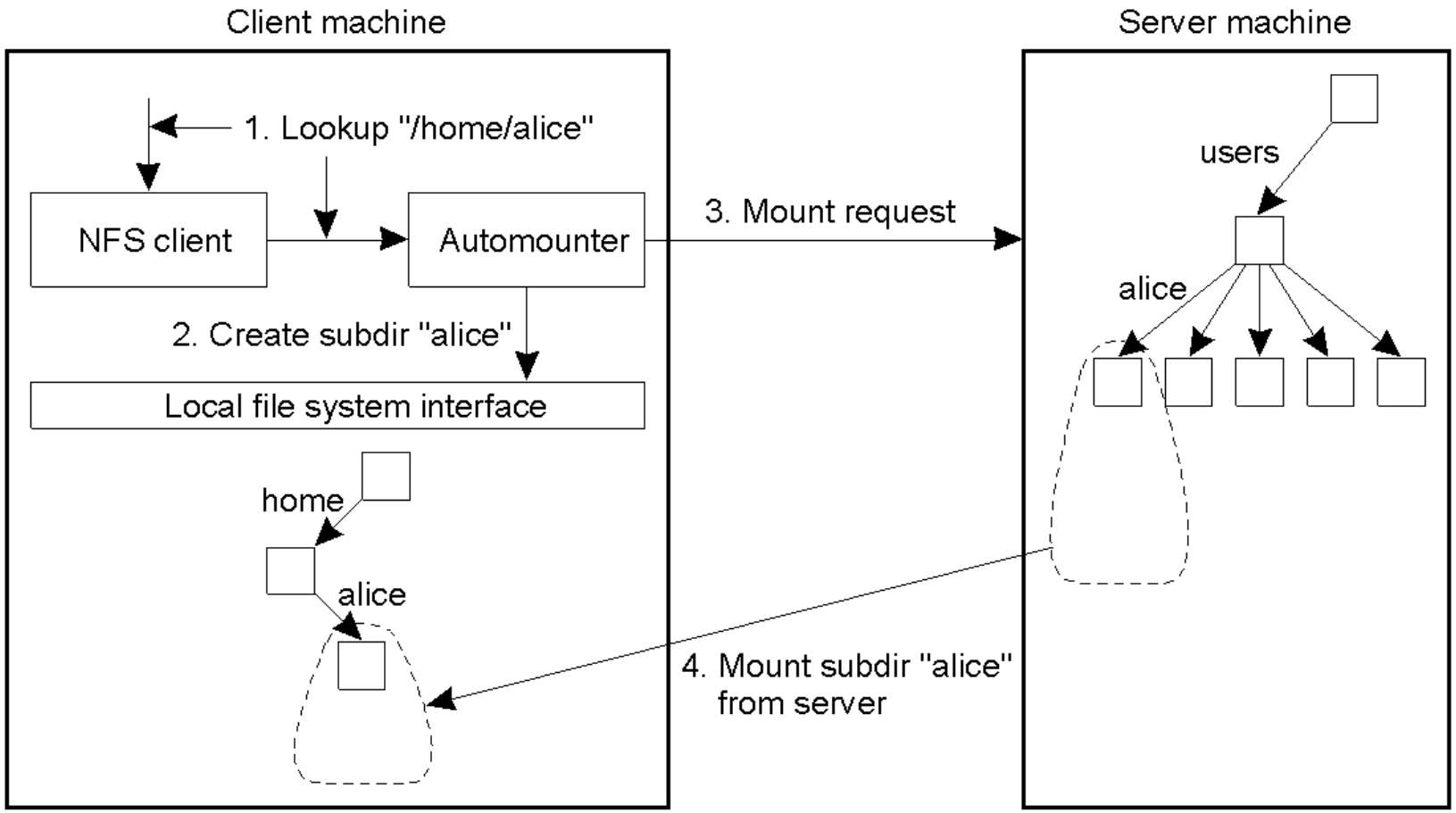
Mounting (part of) a remote file system in NFS.

Naming (2)



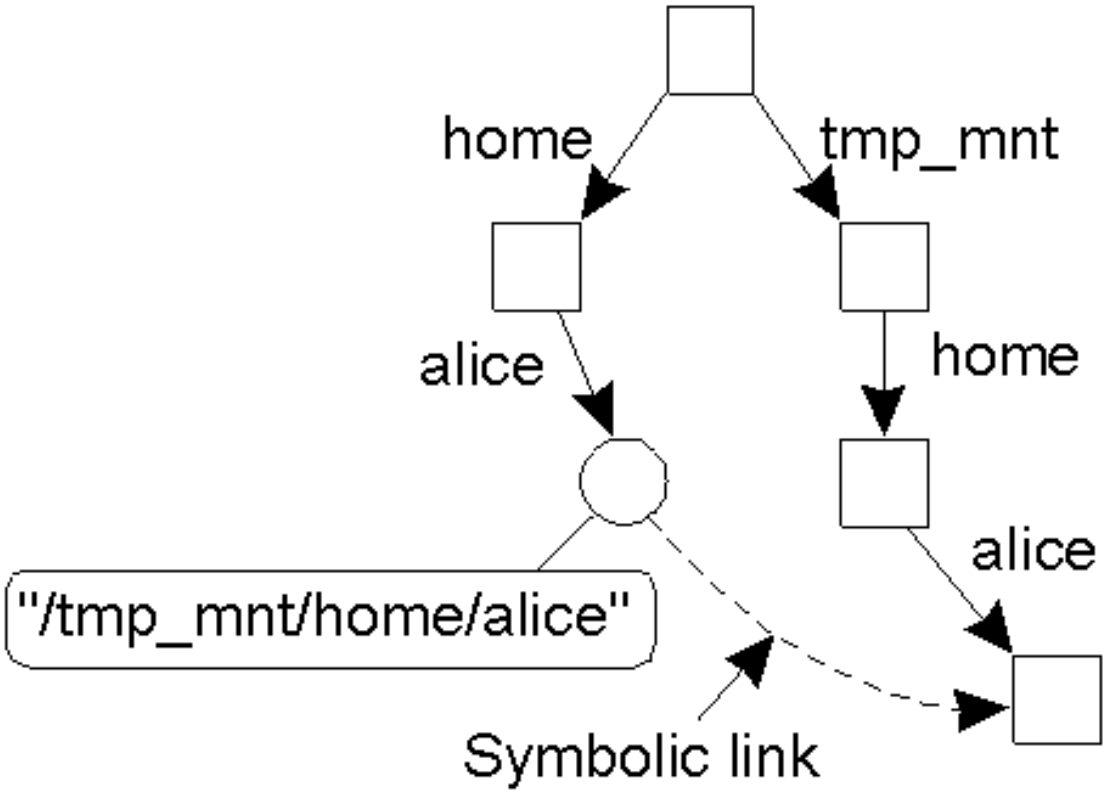
Mounting nested directories from multiple servers in NFS.

Automounting (1)



A simple automounter for NFS.

Automounting (2)



Using symbolic links with automounting.

File Attributes (1)

Attribute	Description
TYPE	The type of the file (regular, directory, symbolic link)
SIZE	The length of the file in bytes
CHANGE	Indicator for a client to see if and/or when the file has changed
FSID	Server-unique identifier of the file's file system

Some general mandatory file attributes in NFS.

File Attributes (2)

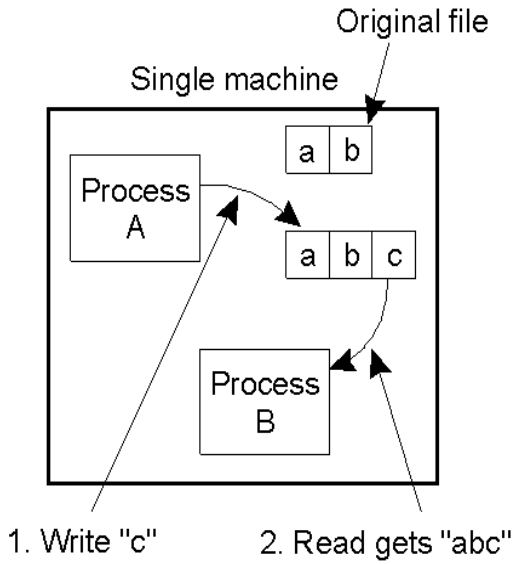
Attribute	Description
ACL	an access control list associated with the file
FILEHANDLE	The server-provided file handle of this file
FILEID	A file-system unique identifier for this file
FS_LOCATIONS	Locations in the network where this file system may be found
OWNER	The character-string name of the file's owner
TIME_ACCESS	Time when the file data were last accessed
TIME_MODIFY	Time when the file data were last modified
TIME_CREATE	Time when the file was created

Some general recommended file attributes.

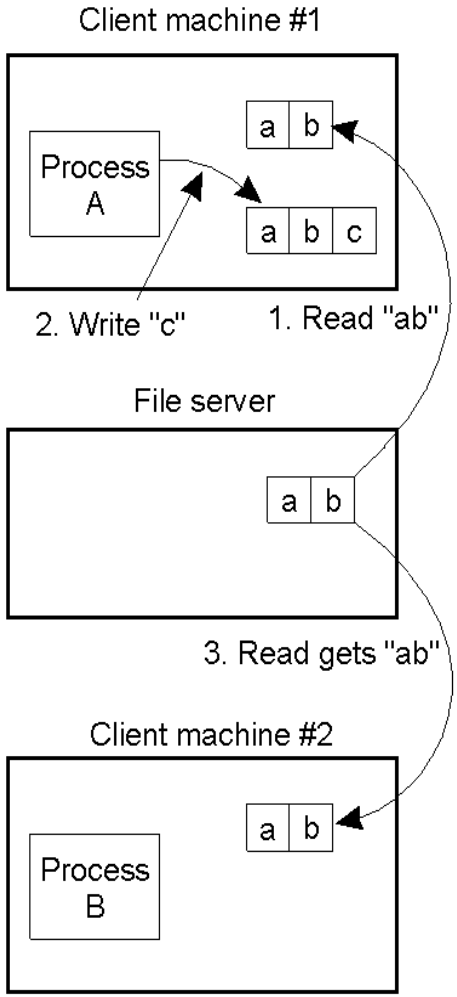
Semantics of File Sharing (1)

a) On a single processor, when a *read* follows a *write*, the value returned by the *read* is the value just written.

b) In a distributed system with caching, obsolete values may be returned.



(a)



(b)

Semantics of File Sharing (2)

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	No updates are possible; simplifies sharing and replication
Transaction	All changes occur atomically

Four ways of dealing with the shared files in a distributed system.

File Locking in NFS (1)

Operation	Description
Lock	Creates a lock for a range of bytes
Lockt	Test whether a conflicting lock has been granted
Locku	Remove a lock from a range of bytes
Renew	Renew the leas on a specified lock

NFS version 4 operations related to file locking.

File Locking in NFS (2)

Current file denial state

Request access	NONE	READ	WRITE	BOTH
READ	Succeed	Fail	Succeed	Succeed
WRITE	Succeed	Succeed	Fail	Succeed
BOTH	Succeed	Succeed	Succeed	Fail

(a)

Requested file denial state

Current access state	NONE	READ	WRITE	BOTH
READ	Succeed	Fail	Succeed	Succeed
WRITE	Succeed	Succeed	Fail	Succeed
BOTH	Succeed	Succeed	Succeed	Fail

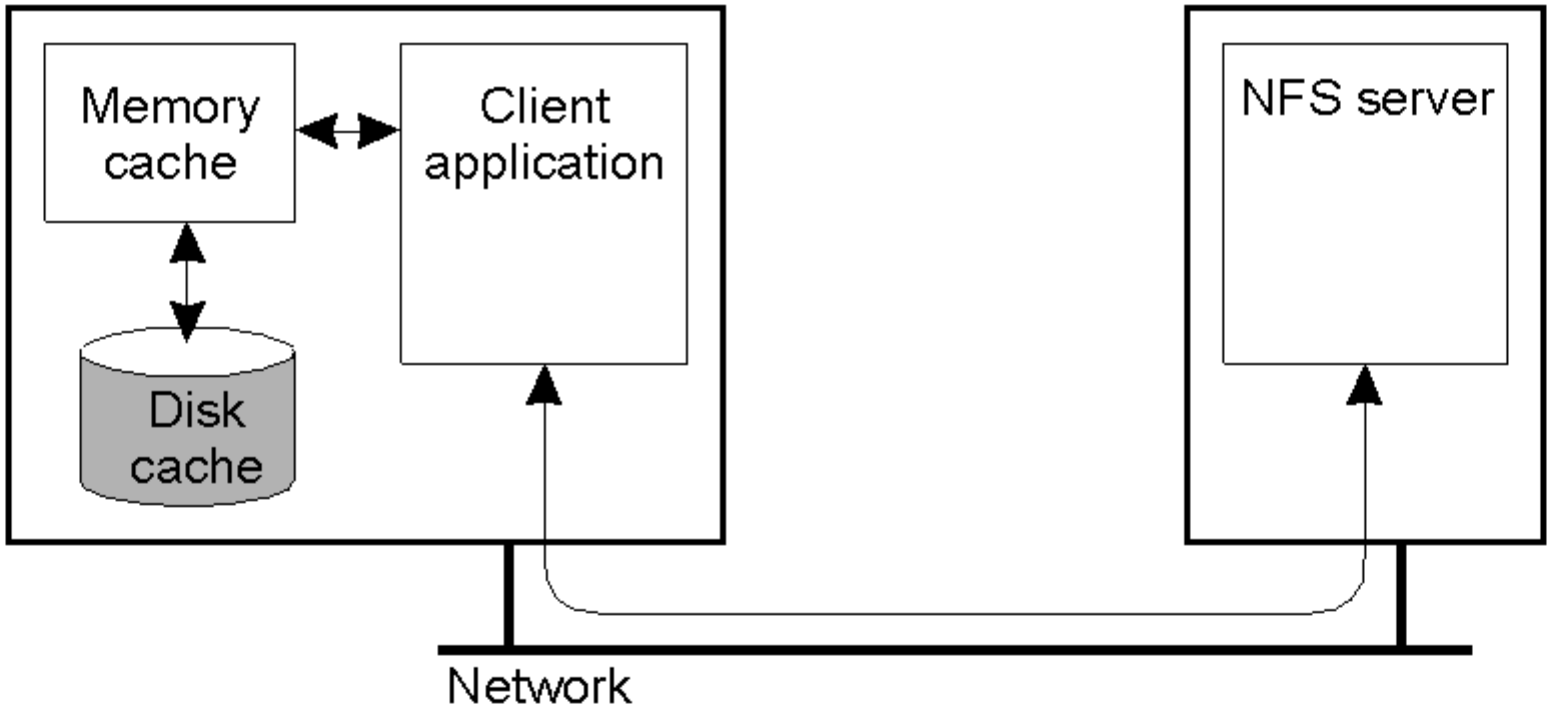
(b)

The result of an *open* operation with share reservations in NFS.

a) When the client requests shared access given the current denial state.

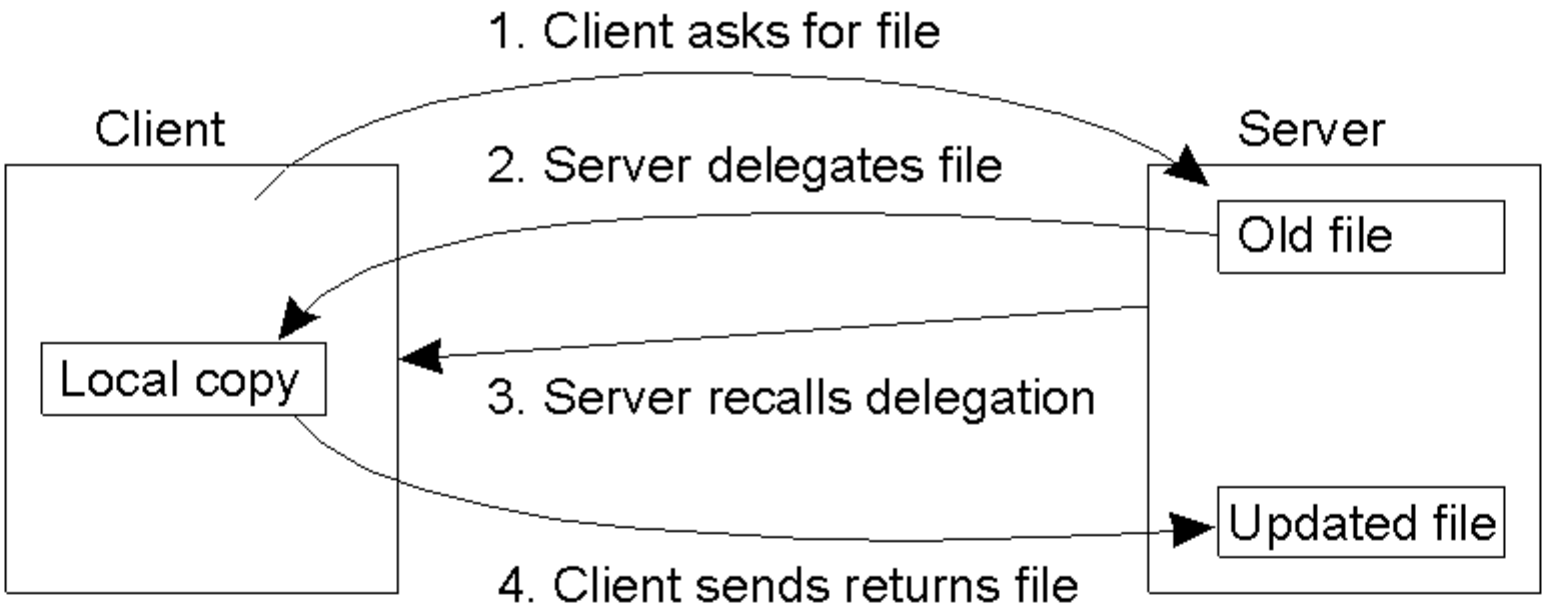
b) When the client requests a denial state given the current file access state.

Client Caching (1)



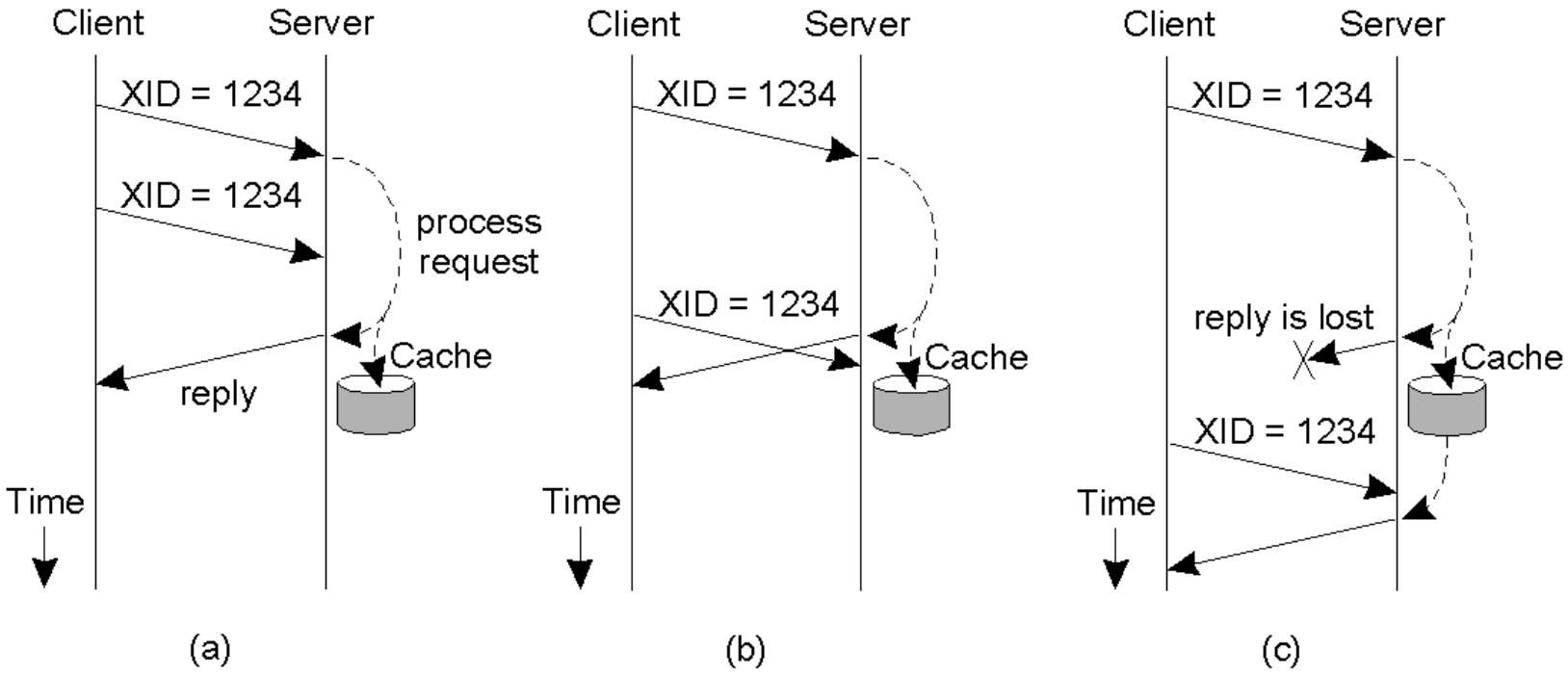
Client-side caching in NFS.

Client Caching (2)



Using the NFS version 4 callback mechanism to recall file delegation.

RPC Failures



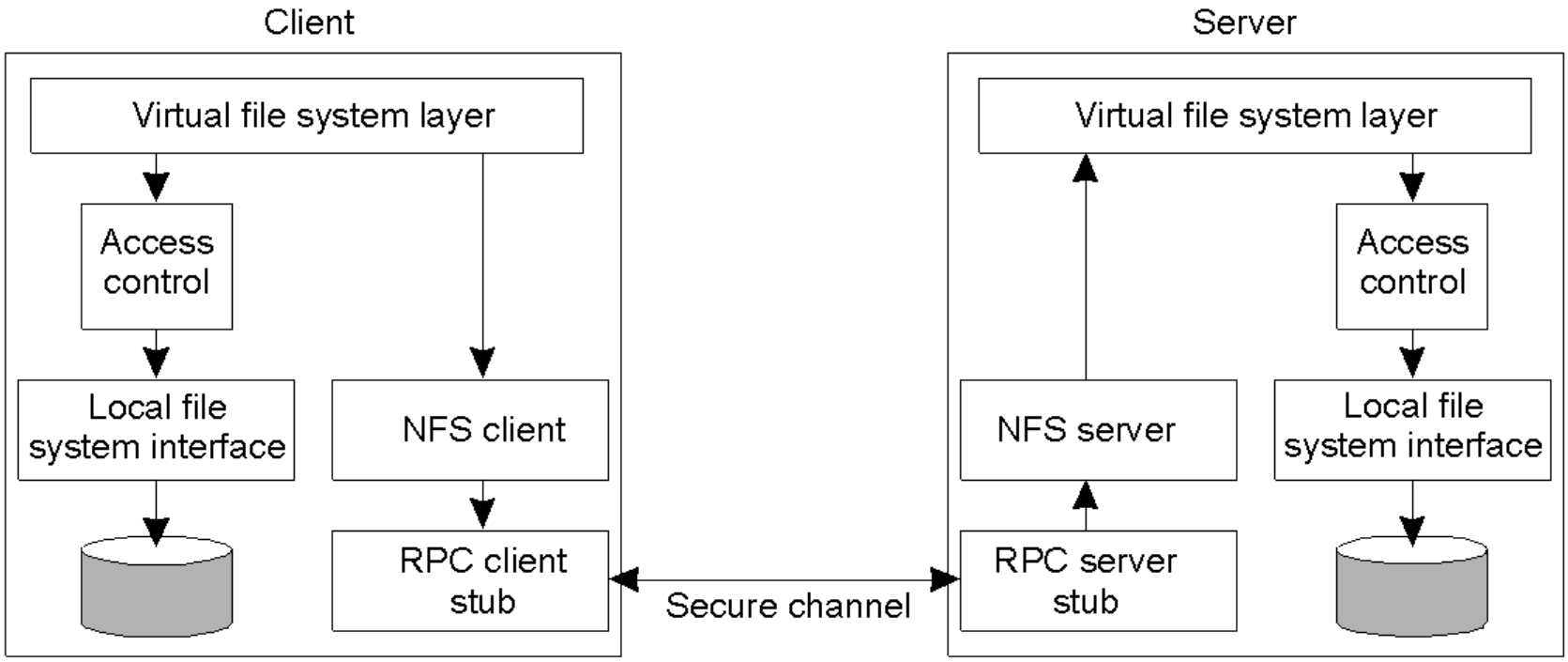
Three situations for handling retransmissions.

a) The request is still in progress

b) The reply has just been returned

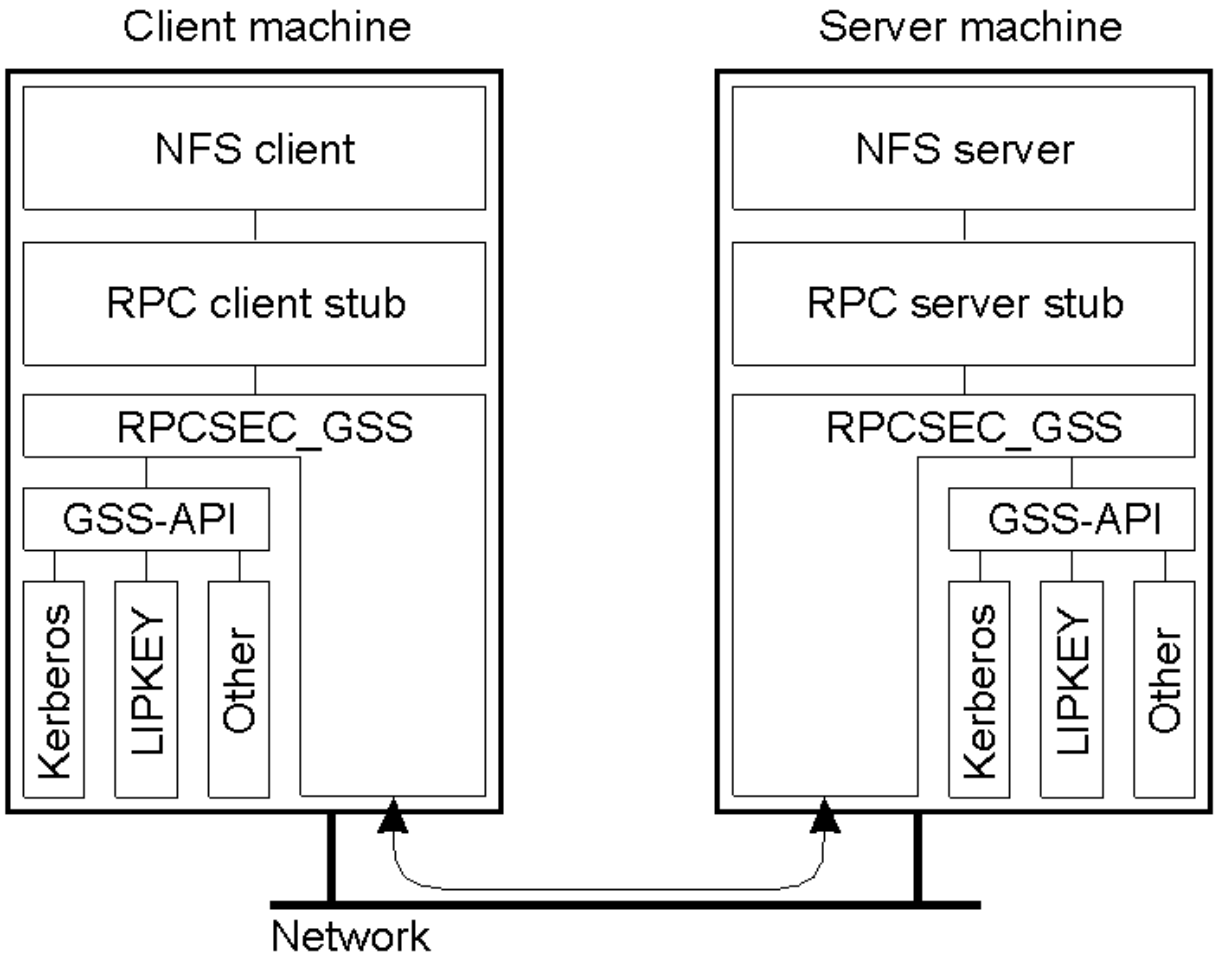
c) The reply has been some time ago, but was lost.

Security



The NFS security architecture.

Secure RPCs



Secure RPC in NFS version 4.

Access Control

Operation	Description
Read_data	Permission to read the data contained in a file
Write_data	Permission to to modify a file's data
Append_data	Permission to to append data to a file
Execute	Permission to to execute a file
List_directory	Permission to to list the contents of a directory
Add_file	Permission to to add a new file t5o a directory
Add_subdirectory	Permission to to create a subdirectory to a directory
Delete	Permission to to delete a file
Delete_child	Permission to to delete a file or directory within a directory
Read_acl	Permission to to read the ACL
Write_acl	Permission to to write the ACL
Read_attributes	The ability to read the other basic attributes of a file
Write_attributes	Permission to to change the other basic attributes of a file
Read_named_attrs	Permission to to read the named attributes of a file
Write_named_attrs	Permission to to write the named attributes of a file
Write_owner	Permission to to change the owner
Synchronize	Permission to to access a file locally at the server with synchronous reads and writes

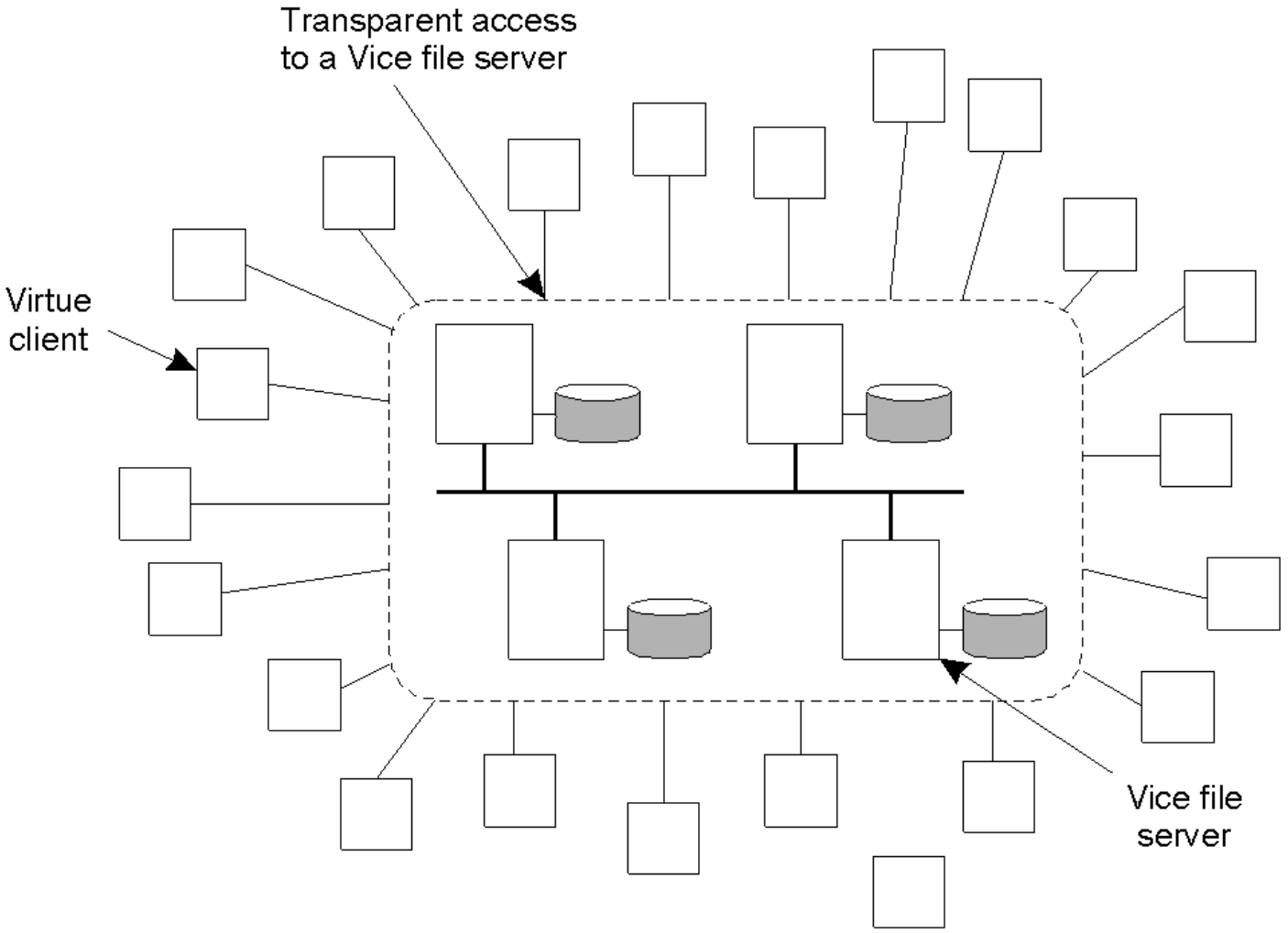
The classification of operations recognized by NFS with respect to access control.

The Coda File System

Type of user	Description
Owner	The owner of a file
Group	The group of users associated with a file
Everyone	Any user of a process
Interactive	Any process accessing the file from an interactive terminal
Network	Any process accessing the file via the network
Dialup	Any process accessing the file through a dialup connection to the server
Batch	Any process accessing the file as part of a batch job
Anonymous	Anyone accessing the file without authentication
Authenticated	Any authenticated user of a process
Service	Any system-defined service process

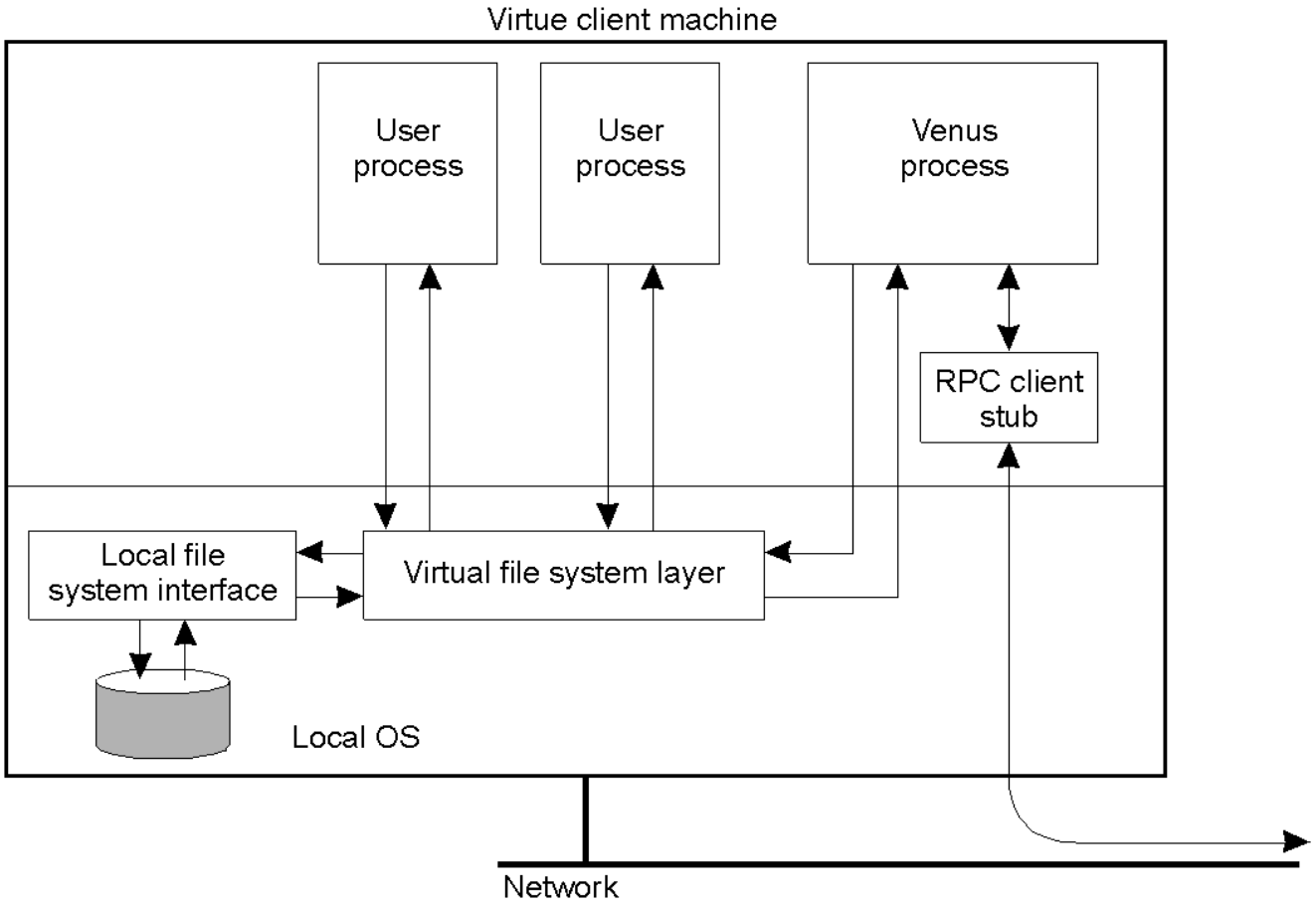
The various kinds of users and processes distinguished by NFS with respect to access control.

Overview of Coda (1)



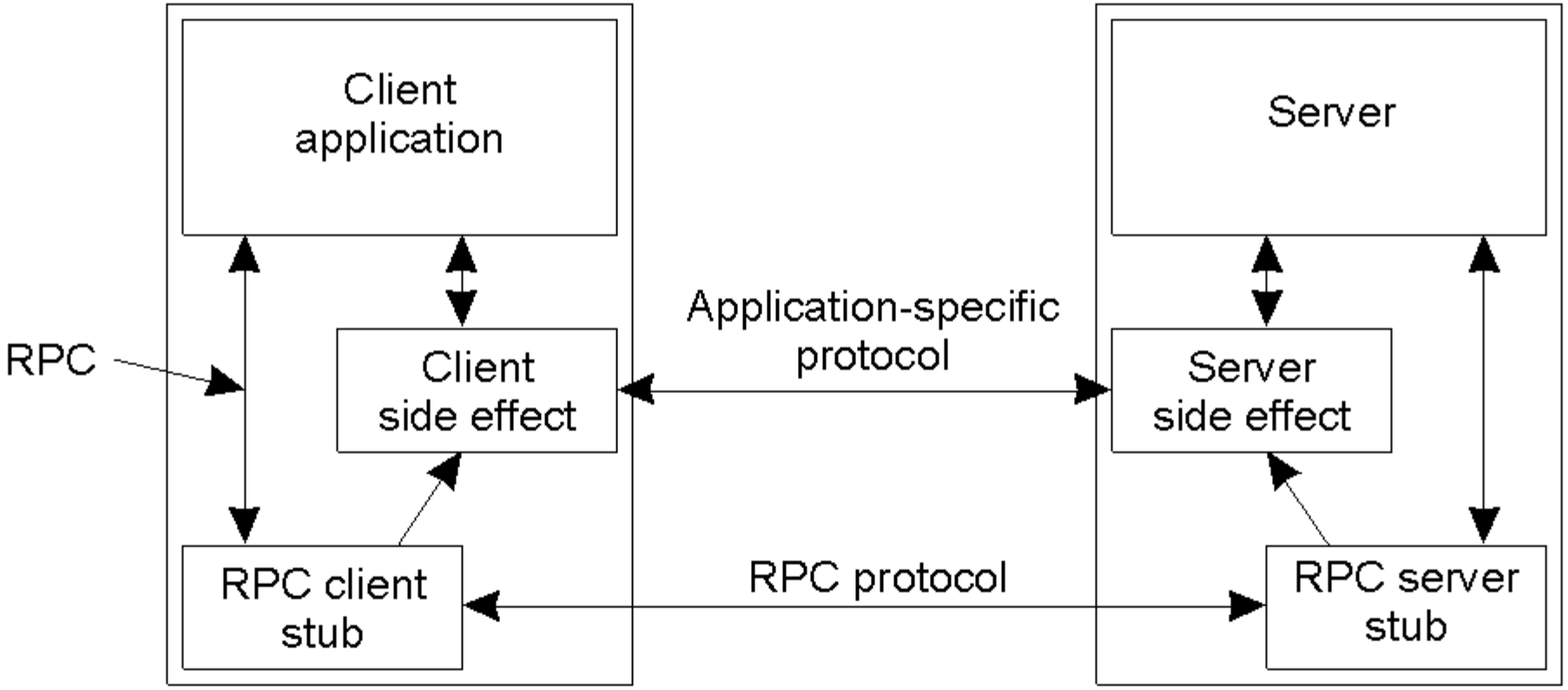
The overall organization of AFS.

Overview of Coda (2)



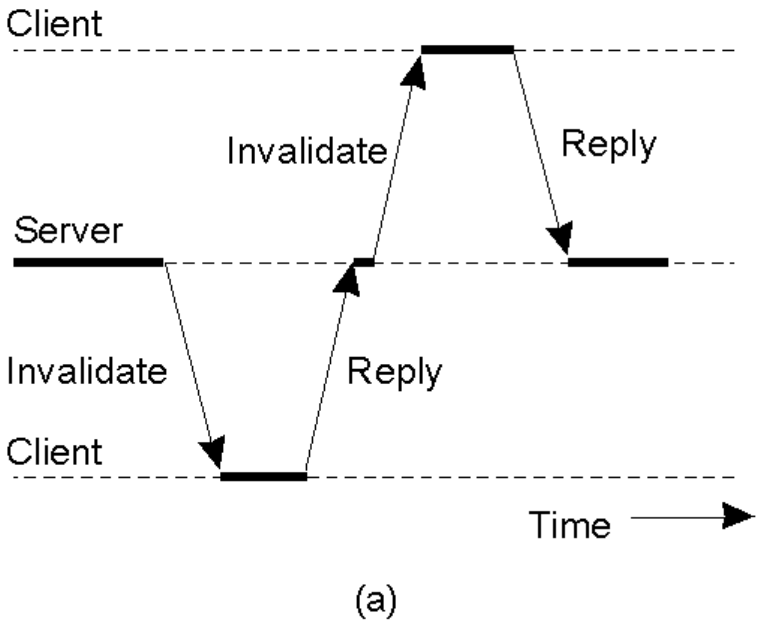
The internal organization of a Virtue workstation.

Communication (1)

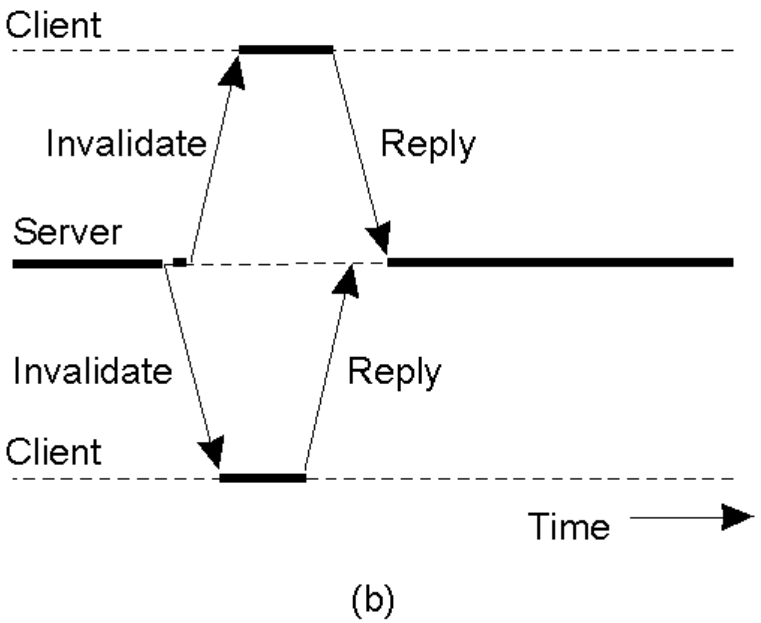


Side effects in Coda's RPC2 system.

Communication (2)



(a)

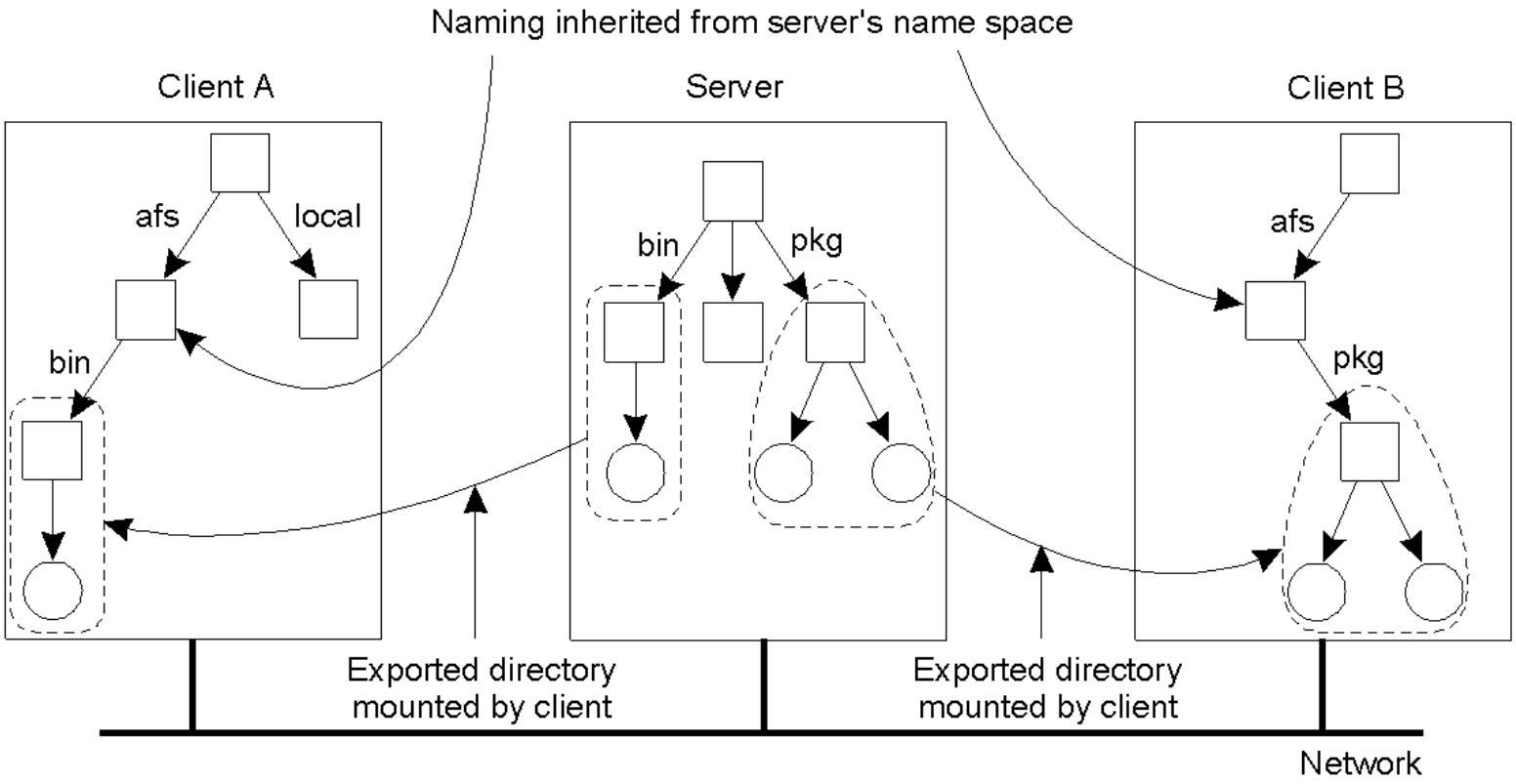


(b)

a) Sending an invalidation message one at a time.

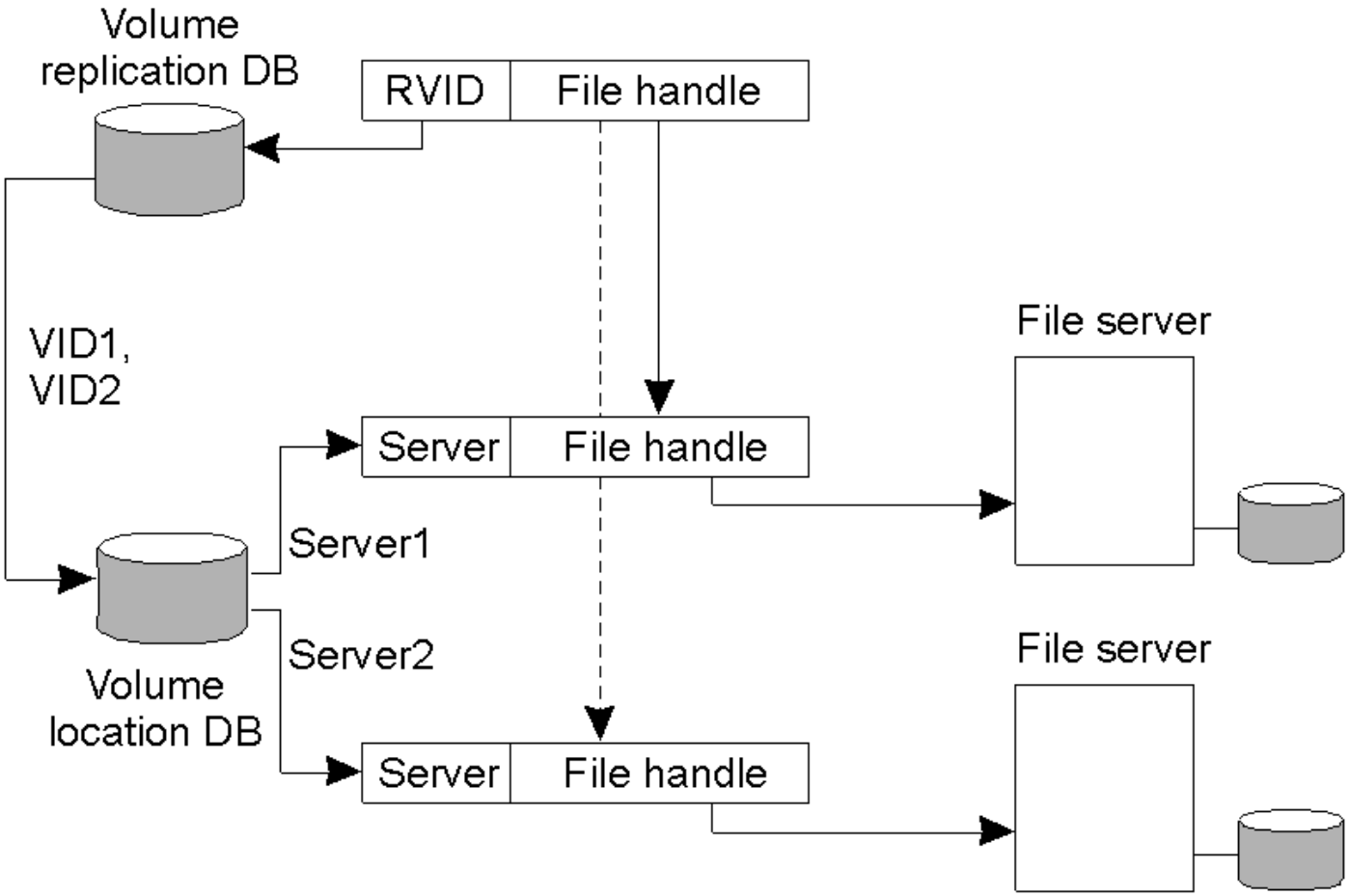
b) Sending invalidation messages in parallel.

Naming



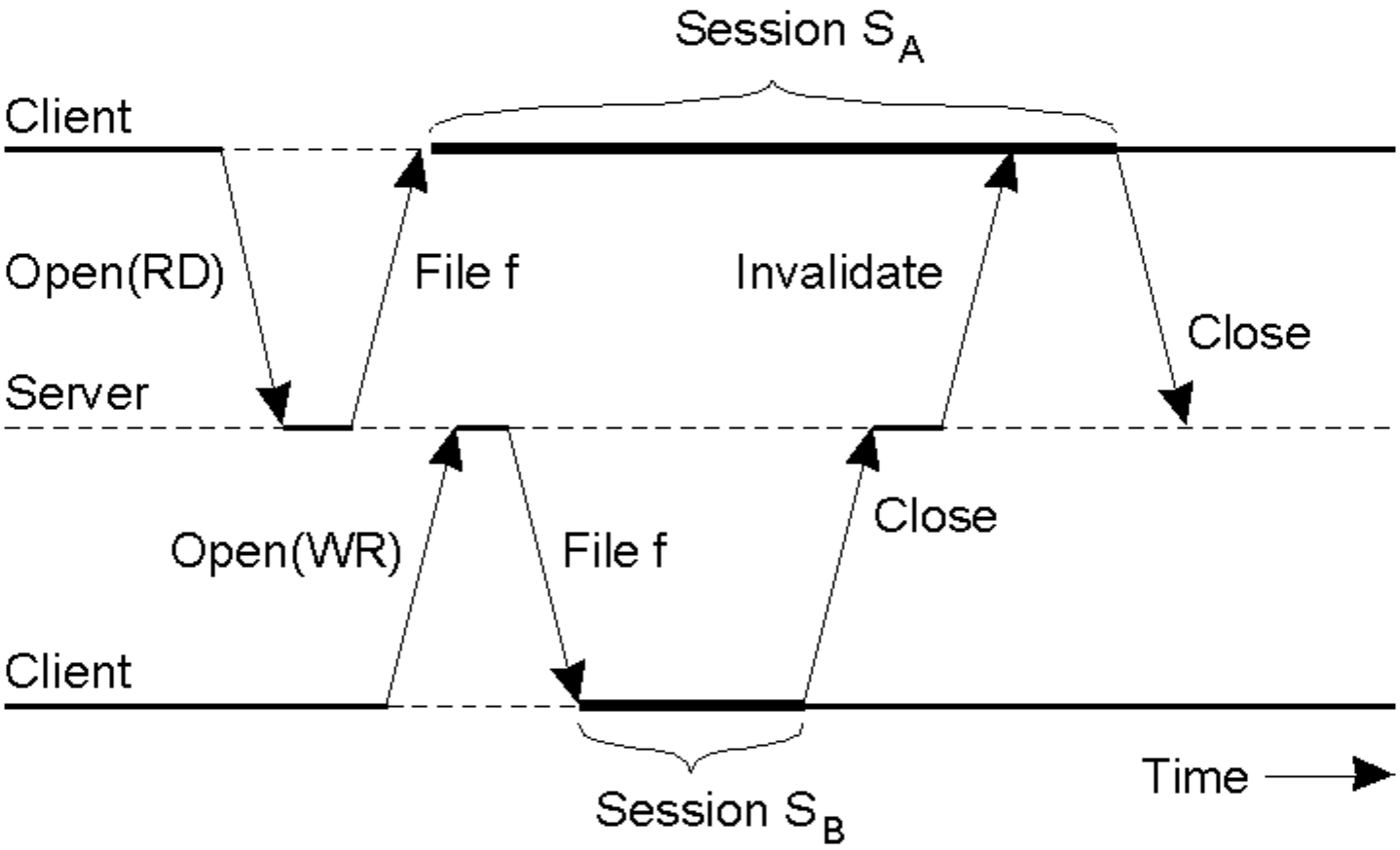
Clients in Coda have access to a single shared name space.

File Identifiers



The implementation and resolution of a Coda file identifier.

Sharing Files in Coda



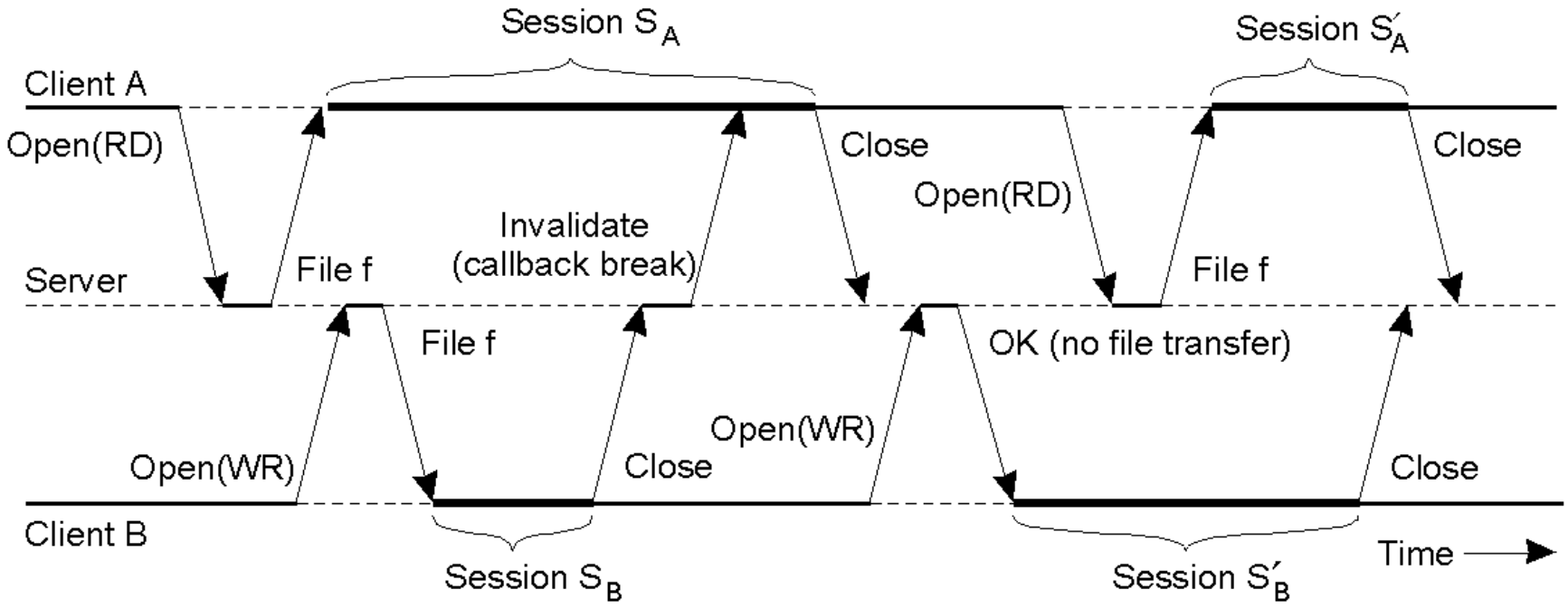
The transactional behavior in sharing files in Coda.

Transactional Semantics

File-associated data	Read?	Modified?
File identifier	Yes	No
Access rights	Yes	No
Last modification time	Yes	Yes
File length	Yes	Yes
File contents	Yes	Yes

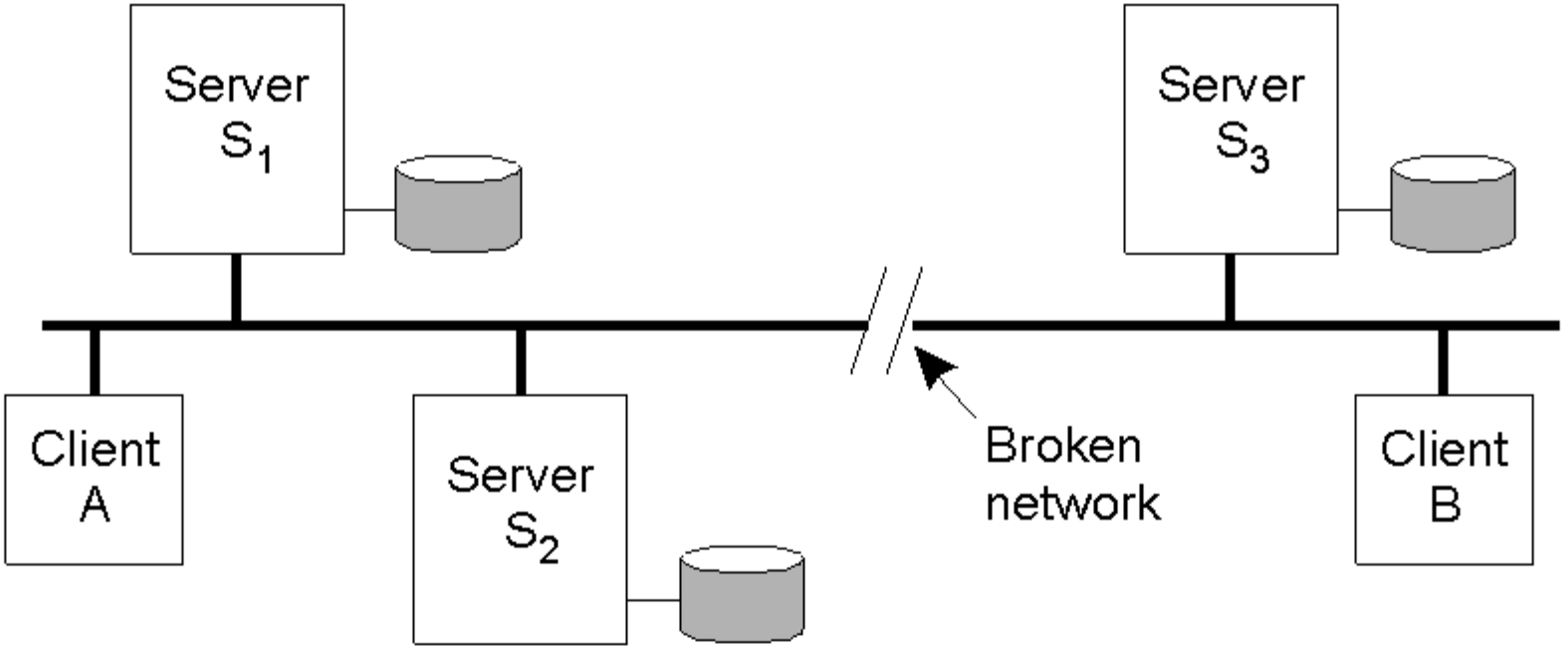
The metadata read and modified for a *store* session type in Coda.

Client Caching



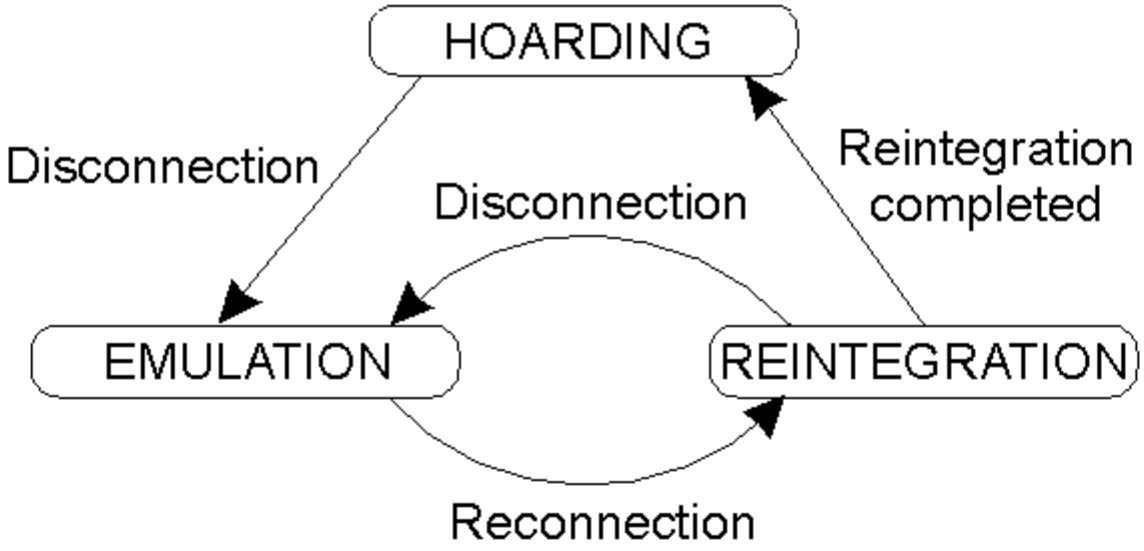
The use of local copies when opening a session in Coda.

Server Replication



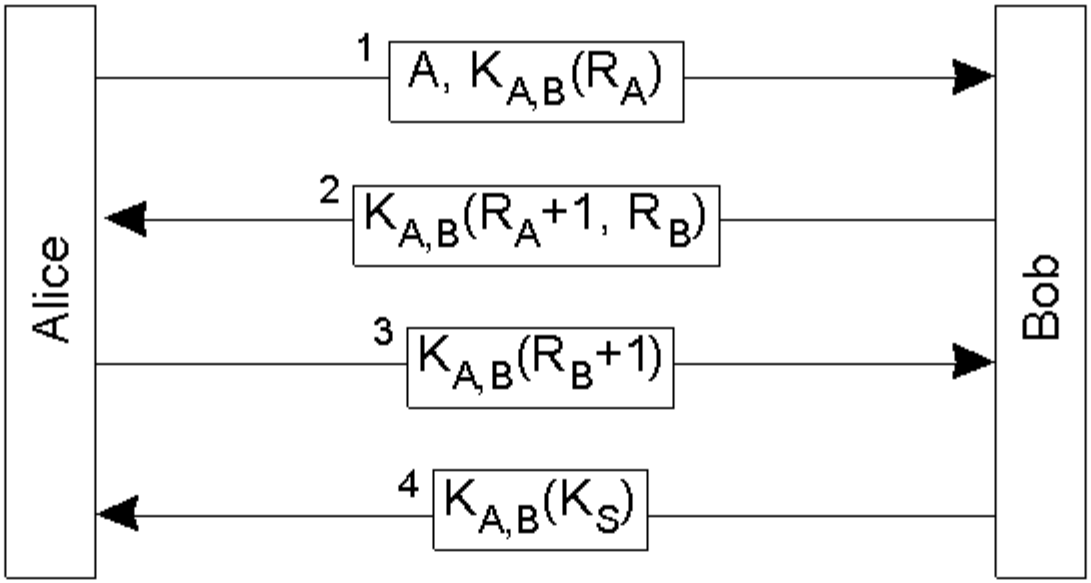
Two clients with different AVSG for the same replicated file.

Disconnected Operation



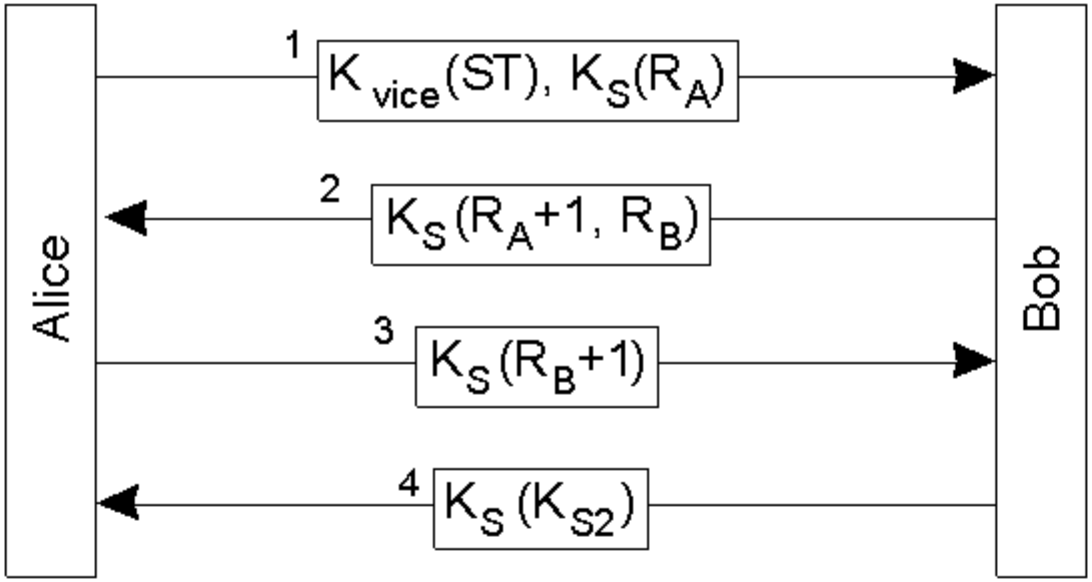
The state-transition diagram of a Coda client with respect to a volume.

Secure Channels (1)



Mutual authentication in RPC2.

Secure Channels (2)



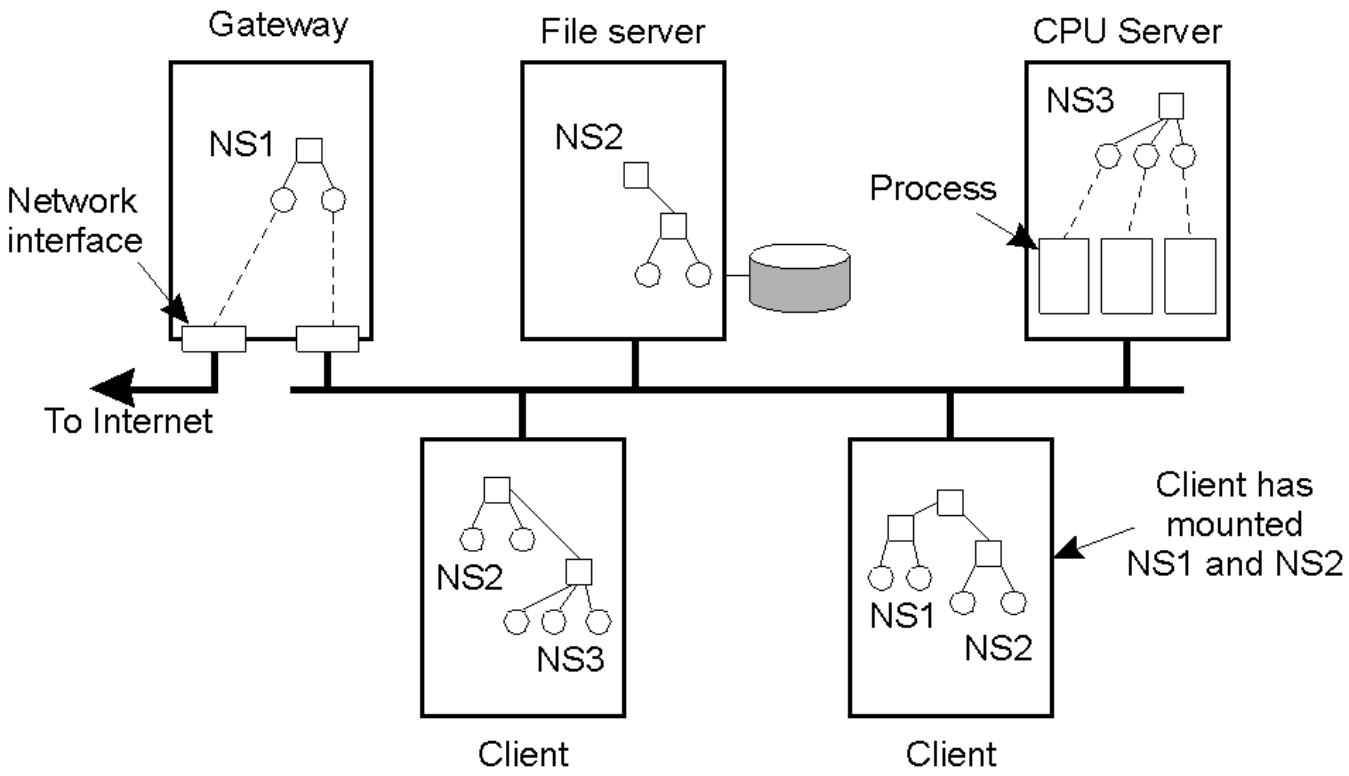
Setting up a secure channel between a (Venus) client and a Vice server in Coda.

Access Control

Operation	Description
Read	Read any file in the directory
Write	Modify any file in the directory
Lookup	Look up the status of any file
Insert	Add a new file to the directory
Delete	Delete an existing file
Administer	Modify the ACL of the directory

Classification of file and directory operations recognized by Coda with respect to access control.

Plan 9: Resources Unified to Files



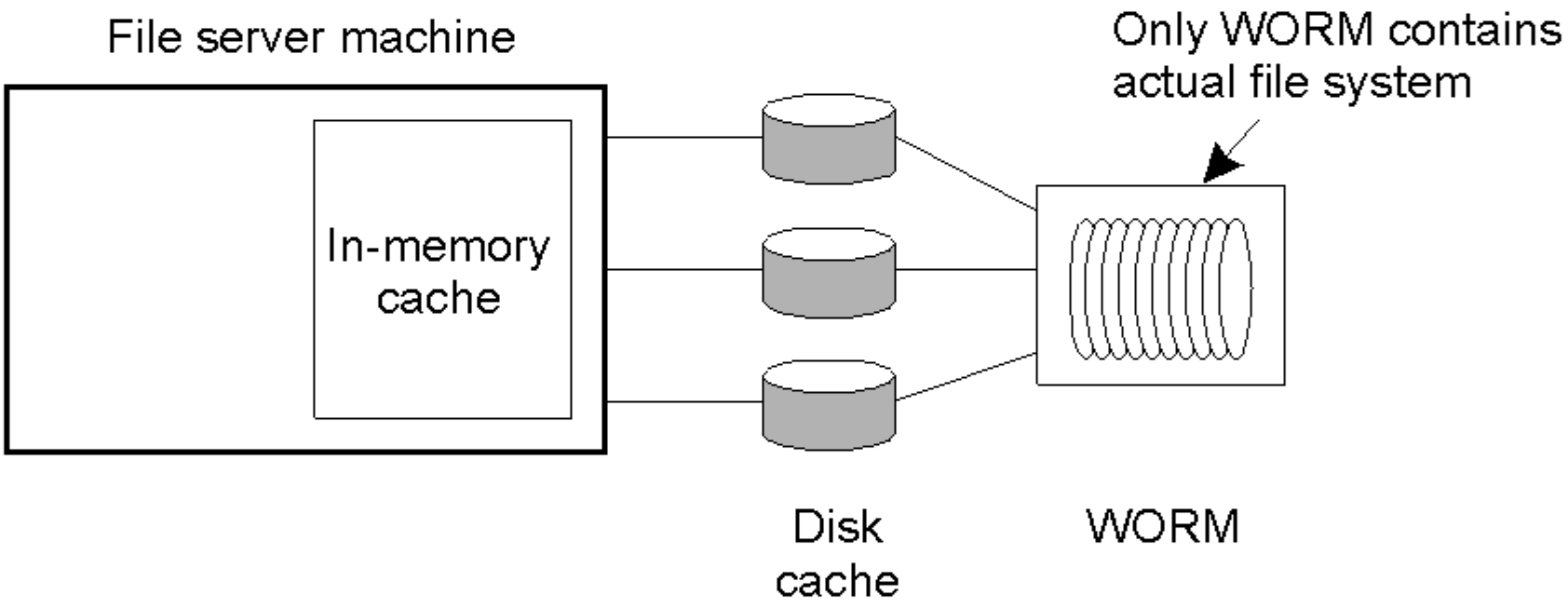
General organization of Plan 9

Communication

File	Description
ctl	Used to write protocol-specific control commands
data	Used to read and write data
listen	Used to accept incoming connection setup requests
local	Provides information on the caller's side of the connection
remote	Provides information on the other side of the connection
status	Provides diagnostic information on the current status of the connection

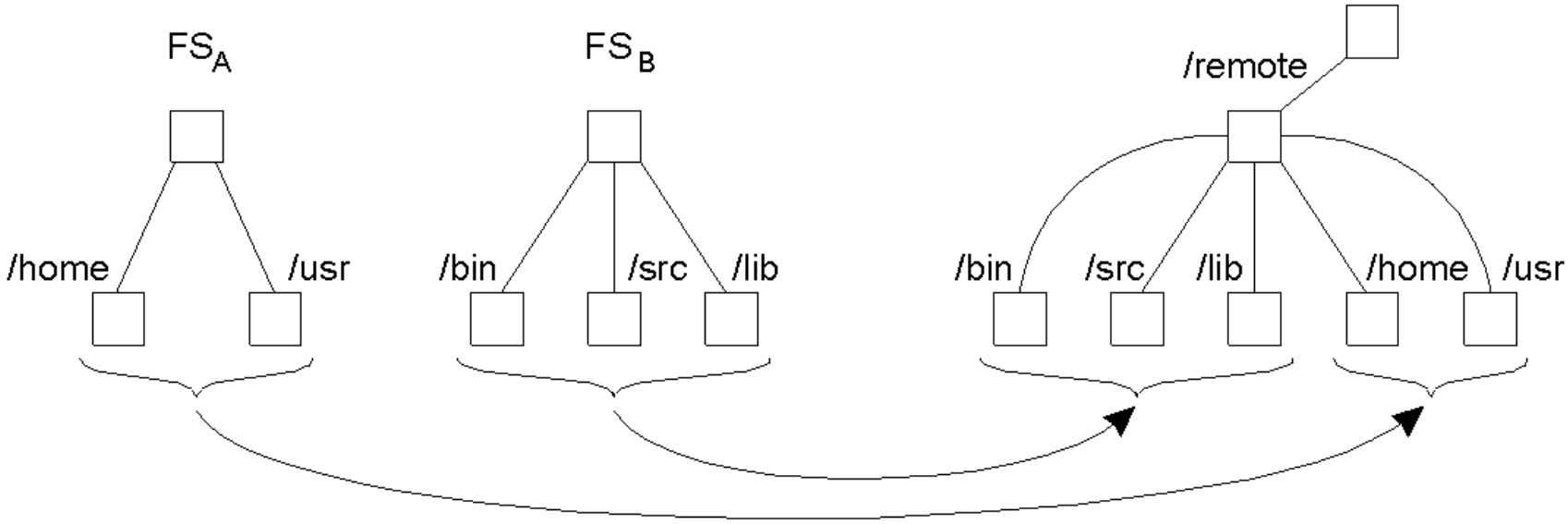
Files associated with a single TCP connection in Plan 9.

Processes



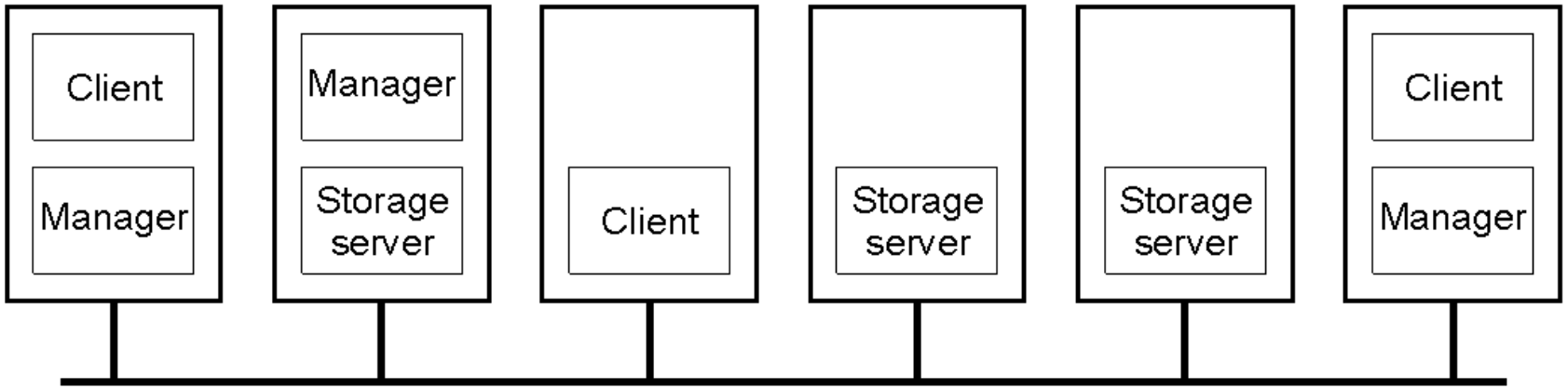
The Plan 9 file server.

Naming



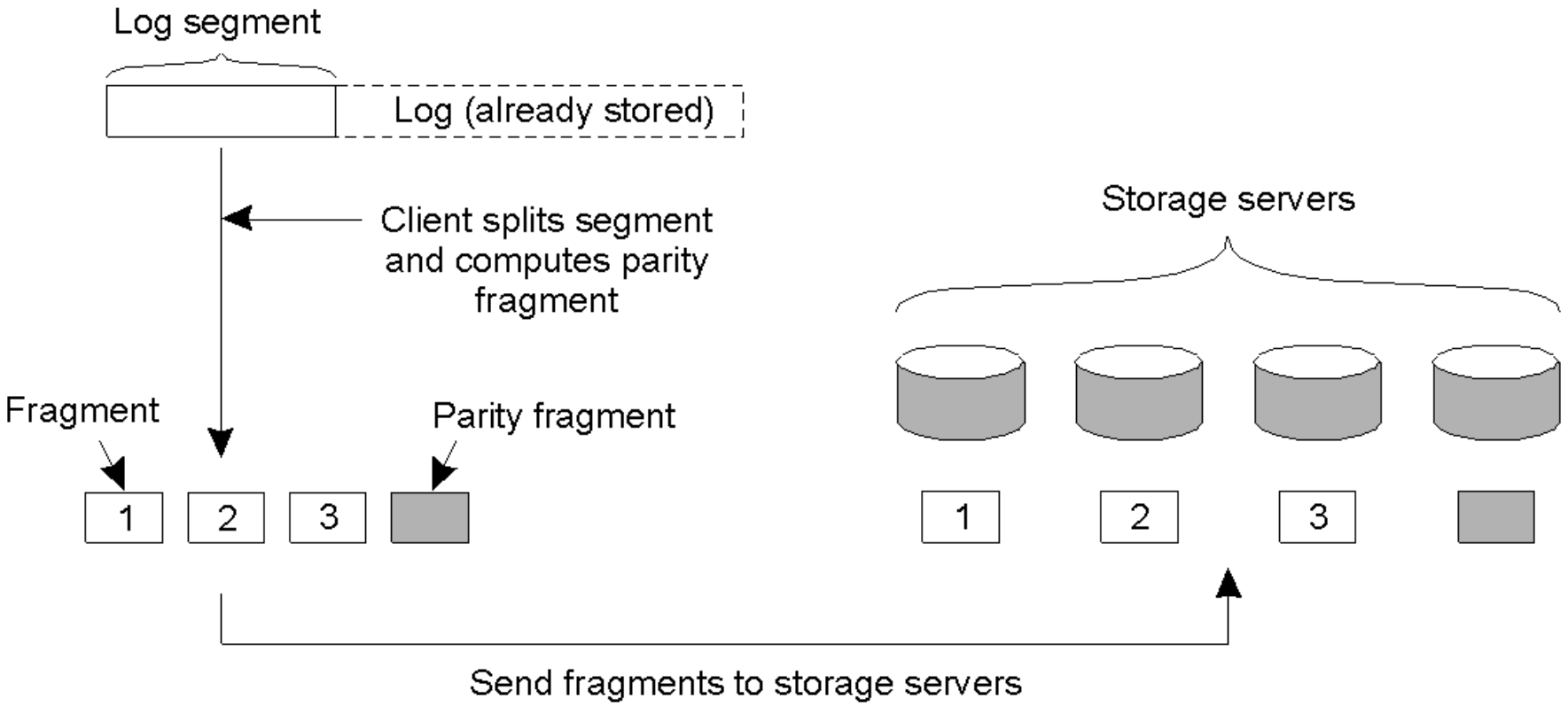
A union directory in Plan 9.

Overview of xFS.



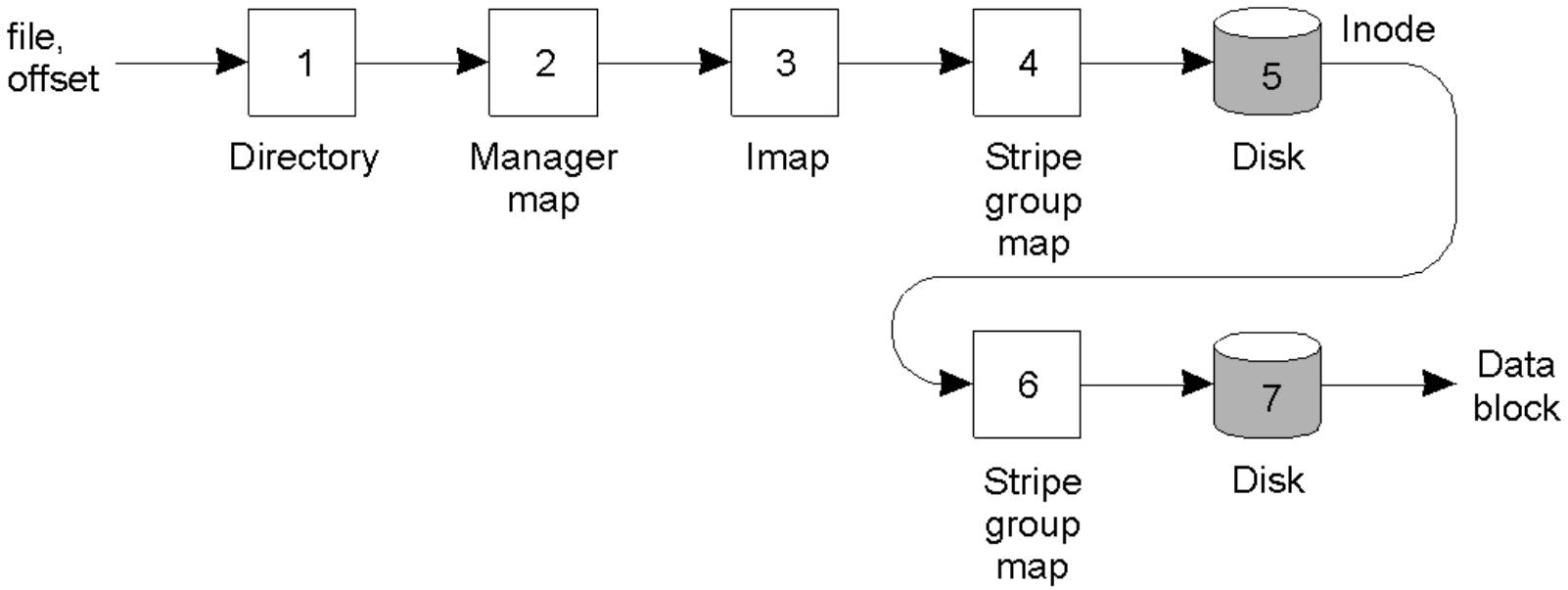
A typical distribution of xFS processes across multiple machines.

Processes (1)



The principle of log-based striping in xFS.

Processes (2)



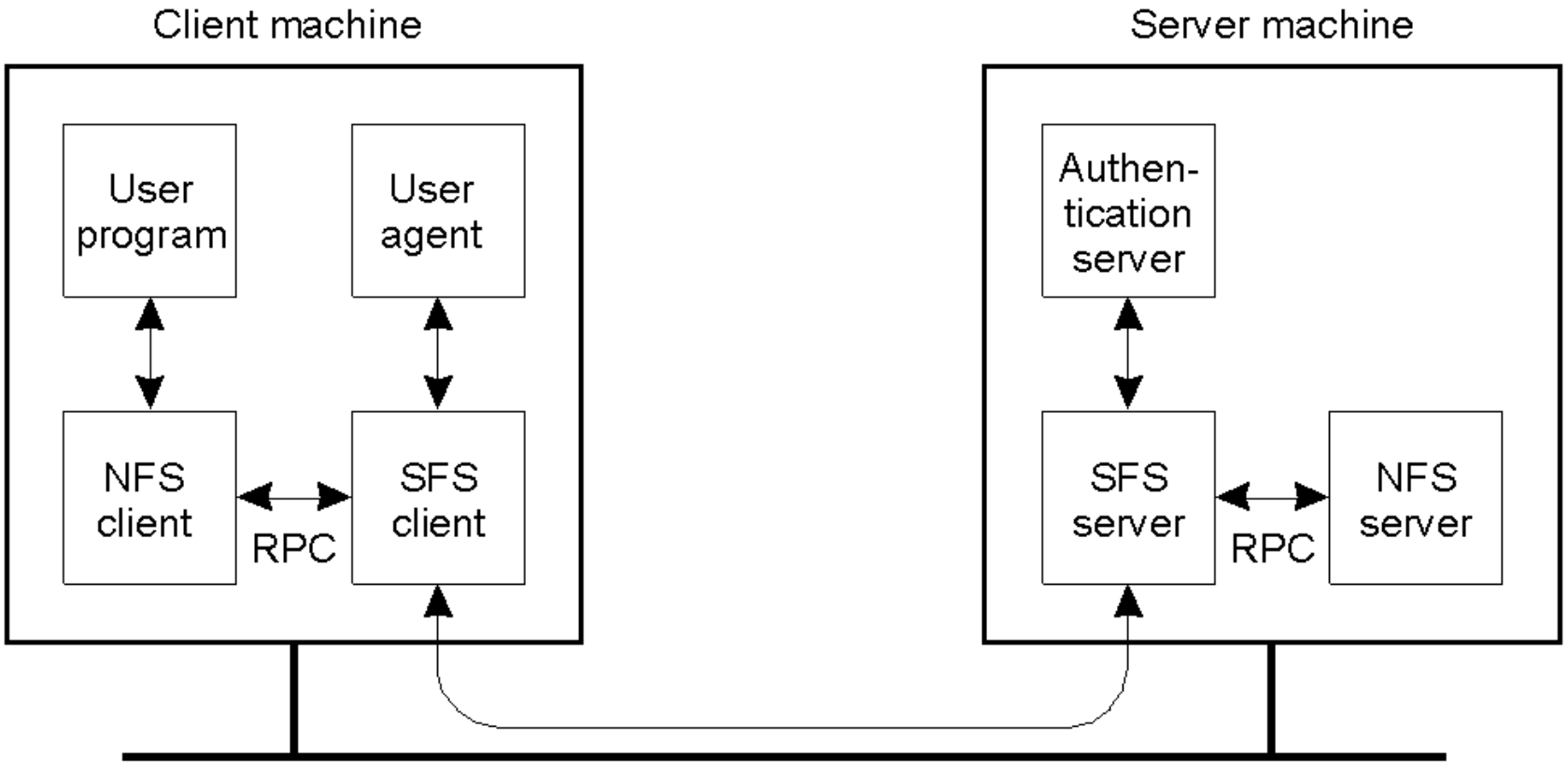
Reading a block of data in xFS.

Naming

Data structure	Description
Manager map	Maps file ID to manager
Imap	Maps file ID to log address of file's inode
Inode	Maps block number (i.e., offset) to log address of block
File identifier	Reference used to index into manager map
File directory	Maps a file name to a file identifier
Log addresses	Triplet of stripe group, ID, segment ID, and segment offset
Stripe group map	Maps stripe group ID to list of storage servers

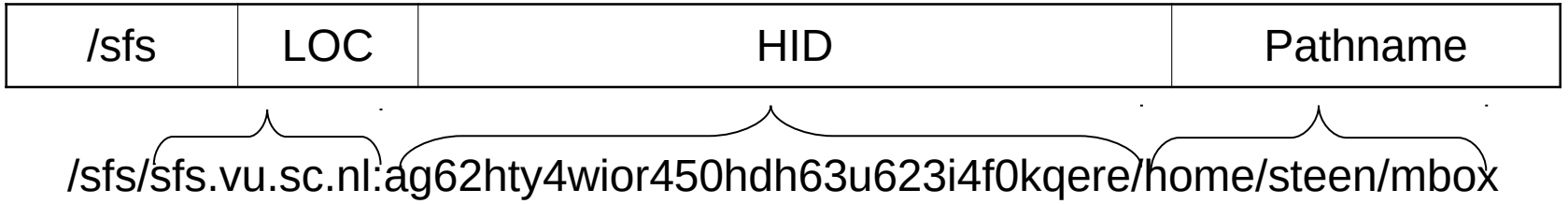
Main data structures used in xFS.

Overview of SFS



The organization of SFS.

Naming



A self-certifying pathname in SFS.

Summary

Issue	NFS	Coda	Plan 9	xFS	SFS
Design goals	Access transparency	High availability	Uniformity	Serverless system	Scalable security
Access model	Remote	Up/Download	Remote	Log-based	Remote
Communication	RPC	RPC	Special	Active msgs	RPC
Client process	Thin/Fat	Fat	Thin	Fat	Medium
Server groups	No	Yes	No	Yes	No
Mount granularity	Directory	File system	File system	File system	Directory
Name space	Per client	Global	Per process	Global	Global
File ID scope	File server	Global	Server	Global	File system
Sharing sem.	Session	Transactional	UNIX	UNIX	N/S
Cache consist.	write-back	write-back	write-through	write-back	write-back
Replication	Minimal	ROWA	None	Striping	None
Fault tolerance	Reliable comm.	Replication and caching	Reliable comm.	Striping	Reliable comm.
Recovery	Client-based	Reintegration	N/S	Checkpoint & write logs	N/S
Secure channels	Existing mechanisms	Needham-Schroeder	Needham-Schroeder	No pathnames	Self-cert.
Access control	Many operations	Directory operations	UNIX based	UNIX based	NFS BASED

A comparison between NFS, Coda, Plan 9, xFS. N/S indicates that nothing has been specified.