

## Progetto di Laboratorio di Sistemi Operativi

### *Descrizione sommaria del progetto*

Il progetto richiesto consiste nella realizzazione di un file storage, quindi con funzionalità simili, ad esempio, a OneDrive, ma con l'utilizzo di UNIX socket. Questo significa che i file non saranno scambiati tramite la rete internet, ma tra processi operanti sullo stesso dispositivo. Inoltre, il server richiesto non deve memorizzare sul disco rigido i file che sta gestendo, ma esclusivamente sulla RAM.

### *Scelte progettuali*

- Se il client, al momento della chiusura della connessione col server, ha ancora dei file aperti, allora i file saranno chiusi automaticamente dal server
- Un file aperto non può essere cancellato
- Un client non può eliminare un file che è aperto da un altro client

### *Descrizione dei dati*

**Impostazioni del server** → il server viene inizializzato attraverso una serie di informazioni memorizzate all'interno di un file in formato ini. Queste informazioni sono relative a:

- Spazio di memorizzazione massimo utilizzabile dal server (**MAX\_STORAGE**)
- Numero massimo di worker threads (**WORKER\_THREADS**)
- Numero massimo di file memorizzabili (**MAX\_STORABLE\_FILES**)
- Il path del file socket utilizzato per interfacciarsi coi client (**SOCK\_PATH**)
- Il path del file di log (**LOG\_PATH**)
- Valore intero compreso tra 1 e 3 (**PRINT\_LOG**) che indica il livello di dettaglio con cui il server dovrà fornire informazioni:
  - 0 → nessuna informazione viene stampata
  - 1 → il server stampa solo le informazioni più importanti (default)
  - 2 → il server stampa tutte le operazioni effettuate

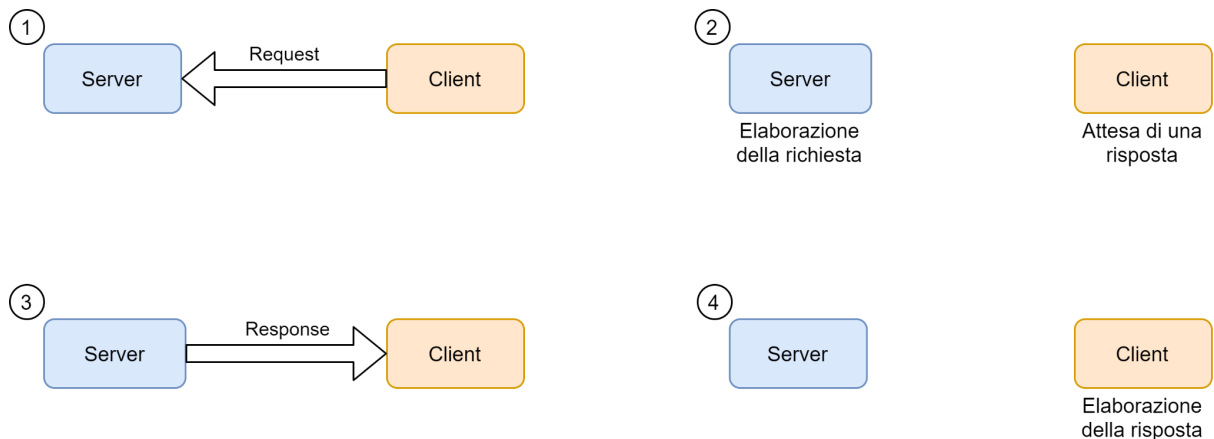
**Memorizzazione dei file** → i file vengono memorizzati attraverso una hash table, che viene allocata interamente dall'inizio, in base a MAX\_STORABLE\_FILES. Questa scelta è dovuta alla volontà di evitare di dover effettuare dei rehashing, che risulterebbero onerosi in termini di calcolo.

Ogni file è gestito dal programma come una struttura dati StoredFile, avente i seguenti campi:

- **char\* path** → il percorso del file
- **size\_t size** → la dimensione del file
- **void\* content** → il contenuto del file
- **list\* pidlist** → la lista dei client che in questo momento hanno il file aperto
- **pthread\_mutex\_t \*mtx** → mutex per gestire l'accesso al file
- **pthread\_cond\_t \*mtx** → variabile condizione del file

**Memorizzazione dei file aperti** → i file aperti vengono anch'essi memorizzati attraverso una hash table. Lo scopo di questa tabella è quello di fornire un accesso immediato ai file aperti da un determinato client, velocizzando di molto operazioni come l'operazione di disconnessione di un client.

Come avviene la comunicazione client-server



Il processo client e il processo server comunicano attraverso uno scambio di almeno due messaggi:

- **Request** → è una stringa con una precisa sintassi  
[ op ] : [ file\_path ] : [ client\_id ] ? [ option ]
  - **op** → tipo dell'operazione richiesta al server
  - **file\_path** → path del file su cui effettuare l'operazione
  - **client\_id** → identificatore univoco del client
  - **option** → può assumere più valori, ognuno con un diverso significato:
    - 'y' indica al server di inviare i file espulsi in caso di capacity misses
    - 'n' indica al server di non inviare i file espulsi in caso di capacity misses
- **Response** → valore intero che fornisce al client informazioni sull'esito dell'esecuzione delle operazioni richieste. Può avere più possibili valori, dichiarati all'interno di myerrno.h:
  - CONNECTION\_REFUSED → la connessione al server è stata rifiutata
  - CONNECTION\_ACCEPTED → la connessione al server è stata accettata
  - EOS\_F → coda FIFO letta correttamente
  - SFILES\_FOUND\_ON\_EXIT → il client ha tentato di chiudere un file aperto anche da altri utenti.
  - S\_STORAGE\_EMPTY → il client ha tentato di accedere a un file, ma il server non ne contiene.
  - SFILE\_ALREADY\_OPENED → il client ha tentato di aprire un file che ha già aperto
  - SFILE\_ALREADY\_EXISTS → il client ha tentato di aprire un file che ha già aperto
  - SFILE\_NOT\_FOUND → non è stato trovato il file richiesto
  - SFILE\_NOT\_OPENED → il client ha provato a compiere operazioni su un file che non ha aperto
  - SFILE\_OPENED → il client ha tentato di cancellare un file aperto
  - SFILE\_TOO\_LARGE → il client ha provato a salvare sul server un file di dimensioni superiori a MAX\_STORAGE
  - S\_STORAGE\_FULL → il server ha raggiunto il numero massimo di file memorizzabili
  - S\_FREE\_ERROR → non è stato possibile liberare spazio per gestire il capacity miss
  - SFILE\_NOT\_EMPTY → il client ha provato a scrivere su un file non vuoto.
  - S\_SUCCESS → la richiesta è stata portata a termine con successo