

Esercizio sulle fifo

Esercizio

Si vuole realizzare una applicazione in ambiente Unix basata su uno schema *cliente-servitore*.

L'applicazione deve essere costituita da due processi (cliente e servitore, non necessariamente parenti):

- il **cliente** è un programma che prevede una sintassi di invocazione del tipo

`cliente com fileout`

Il cliente interagisce con il servitore per richiedere l'esecuzione di un servizio (l'esecuzione del comando `com`) con ridirezione sul file `fileout`.

- il **servitore** è un programma (che non prevede argomenti), che, una volta ricevuti dal cliente `com` e `fileout`, provvede all'esecuzione di `com` con ridirezione su `fileout`.

Impostazione

- Processi non appartenenti alla stessa gerarchia:
 - la comunicazione non può avvenire mediante pipe
 - ➔ utilizziamo una **fifo**
- Il servitore deve essere attivo e pronto ad accettare la richiesta dal cliente.
- Il cliente prende l'iniziativa nella comunicazione.

Soluzione dell'esercizio: struttura del servitore

```
#include <fcntl.h>
#include <signal.h>
#define msgdim 16
int fdin;

main()
{
    int pid, stato; char msg[msgdim], fileout[msgdim];
    if ((fdin=mkfifo("server.in", 0777))<0)
        perror("server- mkfifo in"); /*la fifo esiste già*/
    if ((fdin=open("server.in", O_RDONLY))<0)
    {
        perror("server_open in:");
        exit(-1);
    }
}
```

```
read(fdin,msg,msgdim); /*lettura comando */
read(fdin,fileout,msgdim);/*lettura nome file */
close(fdin);
pid=fork();
if (!pid)
{ close(1);
  fdin=creat(fileout, 0777); /* rid. output */
  execlp(msg, msg, (char *)0);
  perror("execlp");
  exit(-1);}
else
{ wait(&stato);
  printf("stato figlio %d\n", stato>>8);
  exit(0);
}
}
```

Soluzione dell'esercizio: struttura del cliente

```
#include <fcntl.h>
#define msgdim 16
main(int argc, char **argv)
{int fd;char msg[msgdim], fileout[msgdim];
  if (argc!=3){printf("errore sintattico");
  exit(-1);}
  sprintf(msg,"%s",argv[1]);
  sprintf(fileout,"%s", argv[2]);
  if ((fd=open("server.in", O_WRONLY))<0)
  {  perror("client-open fifo in: ");
    exit(-1);
  }
}
```

```
write(fd, msg, msgdim); /* send com */  
write(fd, fileout, msgdim); /* send fileout */  
close(fd);  
  
unlink("server.in");  
}
```

Spunti per estensioni e modifiche

- Generalizzare server:
 - capacità di gestire più richieste
- Estendere lo schema di comunicazione:
 - invece di ridirigere l'output del comando su file, restituire l'output al cliente mediante ridirezione su fifo.