Subiect:  
Să se creeze un server și un client care vor comunica printr-un pipe. Clientul va trimite la server un nume de director, iar serverul îi va răspunde cu numărul de fișiere cu extensia „.txt” din directorul dat sau cu -1 dacă numele trimis de client nu este un director. Se va folosi o comanda shell/script shell. Serverul va rămâne deschis, așteptând cererile clienților.

Barem:  
- 1p – of  
- 1p – creare (conectare) canal de comunicare  
- 1p – comunicare client – server  
- 1p – construire comanda externa  
- 1p – executia comenzii externe  
- 1p – citirea rezultatului comenzii  
- 0.5p – client trimite de n ori (se va specifica – ex: de 10 ori)  
- 0.5p – serverul ruleaza la infinit  
- 1p – serverul trimite rezultatul clientului  
- 2p – clean up

**Rezolvare**

Vom folosi un fișier shell care va verifica dacă am dat un director sau nu și în caz afirmativ să returnăm numărul de fișiere .txt:

[sourcecode lang="bash"]  
#!/bin/sh  
if [ ! -d $1 ]  
then echo “-1″  
exit 1  
fi  
echo `find $1 -maxdepth 1 -iname ‘\*.txt’ | wc – l`  
[/sourcecode]

Fișierul C care face restul este:

[sourcecode lang="C"]  
#include   
#include   
#include   
#include

int main() {  
int pipefp[2],pipepf[2];  
if(pipe(pipepf)==-1) {  
printf(“S-a intamplat o eroare”);  
return 1;  
}  
if(pipe(pipefp)==-1) {  
printf(“S-a intamplat o eroare”);  
return 1;  
}  
int count = 3;  
while (count > 0) {  
int pid = fork();  
if (pid==-1) {  
printf(“Nu s-a putut face forkul”);  
return 1;  
}  
if (pid==0) {  
//client  
//printf(“Child”);  
char fisier[30];  
printf(“Dati numele folderului: :”);  
scanf(“%s”,fisier);  
//printf(“copil: %s %d \n”,fisier,strlen(fisier));  
int nr =write(pipefp[1],fisier,strlen(fisier));  
//printf(“write a scris %d \n”,nr);  
char raspuns[4];  
read(pipepf[0],raspuns,4);  
printf(“Child: Serverul a raspuns: %s \n”,raspuns);  
//printf(“Fiu out”);  
return 0;  
}  
else {  
//server  
//printf(“Parinte \n”);  
char director[30] = “”;  
int re= read(pipefp[0],director,30);  
//printf(“Copilul a trimis %s %d \n”,director,re);  
char comanda[40] = “./shell.sh “;  
strcat(comanda,director);  
//printf(“parinte comanda %s \n”,comanda);  
FILE\* proc = popen(comanda,”r”);  
char rez[4];  
fgets(rez,3,proc);  
//printf(“Rezultat server: %s \n”,rez);  
write(pipepf[1],rez,4);  
//printf(“Iteratie server %s \n”,rez);  
}  
count–;  
}  
close(pipefp[0]);  
close(pipepf[1]);  
close(pipefp[1]);  
close(pipepf[0]);  
return 0;  
}  
[/sourcecode]

Nu merge de o infinitate de ori serverul, dar asta nu se poate realiza în mod practic la pipe și FIFO, doar la coadă de mesaje și memorie partajată.