Sistemes Operatius 1

Sessió de problemes 6 – 15 d'abril del 2020

Introducció

En aquesta sessió de problemes es presenten exercici relacionats amb la tercera pràctica. Aquesta sessió de problemes està centrada en fer servir canonades per comunicar dades entre processos així com els senyals que serveixen perquè processos s'enviïn "interrupcions" entre entre sí. Aquestes interrupcions, però, són un mecanisme gestionat a nivell de programari i no de maquinari.

Aquesta sessió té un exercici que s'ha d'entregar i puntua a la qualificació d'EP. Els problemes que et plantegen així com l'exercici a entregar estan relacionats amb la pràctica a realitzar.

Les canonades

Les canonades és un mecanisme de comunicació entre processos que permet enviar informació d'un procés a un altre. Típicament es fa servir a la línia de comandes, per tal d'enviar la sortida estàndard d'un procés a l'entrada estàndard d'un altre procés.

Les canonades es faran servir aquí (i, en particular, a la pràctica 3) perquè un procés pugui enviar dades a un altre procés. Es faran diversos experiments perquè es pugui entendre bé els conceptes associats.

1. Començarem analitzant la mida de la canonada. Per això farem servir el fitxer `pipe_size.c`, el qual fa que el procés fill escrigui a la canonada. El pare, però, no llegeix cap cada de la canonada. Això farà que la canonada s'ompli. Observa que arriba un moment en què el programa és "penja". Per què és penja? Quina informació ens està donant el programa sobre la mida de la canonada?

2. A Linux existeix una forma d'augmentar la mida de la canonada. Teniu un exemple a `pipe_large.c`. Les crides a sistema que es fan servir per augmentar no formen part de l'estàndard POSIX i, per tant, no tenen perquè funcionar a altres sistemes Unix (com el Mac).

   Proveu el codi si voleu! Pot semblar útil tenir una canonada gran, però no és eficient, atès que una canonada gran fa que es facin servir més recursos del sistema operatiu. A més, existeixen altres sistemes més eficients de comunicació interprocés en cas que es vulgui transferir gran quantitat d'informació (pex comunicació via xarxa, fitxers, fitxers mapats a memòria). Tingueu en compte que la canonada és la forma més "antiga" de comunicació interprocés.

3. A continuació farem que el buffer de la canonada s'ompli pel fill. Després el pare llegirà les dades escrites pel fill. Teniu un exemple al codi `pipe_write_read.c`. Analitzeu el codi, executeu i observeu què fa. Observeu que el fill acaba abans que el pare comença a llegir! A més, veureu que el codi es "penja" al final. Per què?

4. Finalment, farem una prova amb un codi que genera un "flux continu" de dades: el fill hi escriu dades i el pare les llegeix. Teniu el codi al fitxer `pipe_write_read_continuous.c`. Aquest darrer exemple és la forma en què es comuniquen dos processos quan generem una canonada.

Es proposen a continuació un parell d'exercicis, relacionats amb la pràctica, que fan servir les canonades.

1. Al fitxer pipe_size.c hem vist la mida, en bytes, del buffer associat a la canonada. Modifiqueu el codi per escriure-hi valors sencers. Aneu incrementant el valor que hi escriviu al buffer. Quants valors sencers hi podeu escriure?

2. Agafeu el fitxer `pipe_write_read.c` perquè el fill hi escrigui sencers en format binari (i.e. fent servir les funcions read i write). Aneu incrementant el valor que hi escriviu al buffer. Un cop el buffer sigui ple, el pare començarà a llegir els valors. Imprimiu els valors llegits per pantalla per assegurar que els valors que llegiu són els correctes.

Els senyals

Els senyals són un mecanisme de comunicació molt senzill entre processos. Són interrupcions de programari que dos processos poden enviar entre sí per senyalitzar que ha tingut lloc un esdeveniment. Vegem-ne un parell d'exemples

1. El fitxer `sigterm_sigint.c` conté un exemple en què l'aplicació espera a rebre un senyal. Observeu el codi i indiqueu quin és el comportament d'aquest: quan imprimirà el missatge "Exiting main"?

2. El fitxer `sigprocesses.c` conté un segon exemple en què dos processos, pare i fill, s'envien senyals entre sí. Aquest és un mecanisme de sincronització senzill entre dos processos (a l'actualitat hi ha mecanismes de sincronització més avançats que es veuran a Sistemes Operatius 2 com, per exemple, els semàfors). Com sap el pare quin és el seu fill? Com sap el fill quin és el seu pare?

Exercici a entregar

Es proposa a continuació un exercici que forma part de les activitats presencials (AP) de l'assignatura. Els detalls de l'avaluació s'indiquen al final.

En concret, el codi ha de fer fer les següents tasques

1. El procés pare crea un procès fill. El procés pare i el procés fill es comunicaran entre sí mitjançant una canonada.
2. El procés pare genera N valors sencers aleatoris i els escriu a la canonada a mesura que els va generant. Per fer-ho, escriu primer el valor d'N i després els valors sencers aleatoris (podeu fer servir la funció C "rand" per generar nombres aleatoris).
3. El procés fill, mentrestant, s'espera a rebre un senyal del pare.
4. Així que el procés pare ha escrit els N valors a la canonada envia el senyal SIGUSR2 al procés fill.
5. En rebre el procés fill el senyal, aquest llegeix primer el valor d'N (que li indica el nombre de valors sencers escrits) així com els valors sencers que hi ha a continuació. El fill realitzarà la suma dels valors sencers que hi ha a la canonada.
6. Mentre el procés fill realitza la suma el procés pare s'espera.
7. Un cop el procés fill ha realitzat la suma escriu el resultat a la canonada i envia el senyal SIGUSR1 al pare.
8. El pare, en rebre el senyal, llegeix de la canonada el valor de la suma i l'imprimeix per pantalla.
9. Els dos processos finalitzen contents i feliços!

Per fer l'exercici es proporciona una plantilla que serà d'ajut per fer-lo. A la plantilla es fa servir un valor de N=1000. Podeu fer servir un altre valor d'N si voleu. Es recomana fer servir un valor d'N prou gran perquè es trigui un "cert temps" per fer la inserció i lectura de les dades. De la mateixa forma, es recomana que sigui prou petit perquè l'aplicació no es quedi penjada en inserir les dades a la canonada (recordeu que el buffer de la canonada té una mida relativament petita; veure els exercicis de la canonada).

Avaluació de l'exercici

L'avaluació d'aquest exercici només tindrà els valors 10, 5 i 0. Per poder obtenir la qualificació de 10 cal entregar l'exercici i la rúbrica. A més, l'exercici ha de funcionar correctament. En cas que l'aplicació no funcioni o la rúbrica corresponent no ho reflecteixi, la qualificació serà de 5. Finalment, en cas que no s'entregui l'exercici o la rúbrica, o la rúbrica no indiqui perquè el codi avaluat no funciona, o la solució entregada no tingui res a veure amb l'exercici proposat l'avaluació serà de zero.

Es demana entregar, en grups de dues persones, l'exercici proposat aquí a través del campus. S'entregarà fent servir el número de grup que se us ha assignat a classe presencial. Així, per exemple, si sou el grup 31 entregueu l'exercici amb un únic fitxer que s'anomeni grup31.c. Eviteu posar noms a l'exercici per assegurar l'anonimat de l'entrega.

Hi ha tot el dia d'avui, dia 15 de d'abril, per entregar l'exercici. Dijous 16 es faran públics al matí els exercicis entregats a l'enllaç següent

https://drive.google.com/drive/folders/1S594dAEbYW02gYdnpAU2oXm3sYnfTcYv

Cada grup d'estudiants haurà d'avaluar dos exercicis diferents: en particular, cada grup avaluarà els dos exercicis amb un número de grup immediatament inferior al que tinguin assignats ell@s. El grup 31 haurà d'avaluar doncs els grups 30 i 29 (o els que hi hagi immediatament inferiors).

L'avaluació es farà fent servir la rúbrica que hi ha disponible al campus. Avaluar cada exercici no hauria de portar més de 15 minuts. Cada grup entregarà les seves dues rúbriques via campus. La data termini per entregar les dues rúbriques assignades és el proper divendres.

L'avaluació assignada pels vostres companys es farà servir per decidir quines solucions es mostren a la propera sessió de problemes.