Los procesos (2)

Sistemes Operatius I

Oliver Díaz

Grau d'Enginyeria Informàtica

Que veremos hoy

- Test de Evaluación Formativa (TA)
- Resolución test / Repaso

Test

QÜESTIONARI

Test sobre unitat 3: Els procesos

Aquesta prova consta de **5 preguntes tipus test**. Cada pregunta té **una única resposta correcta**. Les respostes incorrectes NO resten punts. Un cop començat, disposareu de **25 minuts** per contestar a les preguntes i només es podrà fer un **únic intent**.

0 de 92 intentades **Data límit 22 de març 2022**


11:15-11:40; Campus Virtual

Asistencia

Formulario Google

CV – Teoria (Presencial) o QR

Test

- **¿Habia alguna pregunta trampa, no?**
- **Bueno, veámoslo de nuevo en detalle**

Pregunta 1

- **Les característiques de les funcions fork i exec són:**

  a) La funció exec sempre s'ha d'utilitzar en combinació amb la funció fork.

  b) La funció fork crea una còpia del procés pare i la funció exec crea un nou procés independent.

  c) La funció fork crea una còpia del procés pare i la funció exec s'utilitza només per especificar els arguments d'execució a passar al procés fill.

  d) La funció fork crea una còpia del procés pare i la funció exec permet reemplaçar la imatge del procés fill per una altra.

Pregunta 1

- `fork()`. Crea un nuevo proceso, cópia del proceso padre, pero independiente de éste.
- `exec()`. Famílias de funciones que reemplazan la imagen del proceso con la especificada a exec.

Fork-bomb

```
0 #include <stdio.h>
1 #include <unistd.h>
2
3 int main(void)
4 {
5     while (1) { fork(); }
6 }
7
```

Pregunta 2

- **Quina és l'avantatge que a UNIX, a diferència de Windows, s'hagi dividit l'execució de nous processos en dos passos: les crides a sistemes fork i exec.**

  a) Només és útil per implementar canonades i redirecció.

  b) Permet gestionar, de forma senzilla, tot el context en què s'executarà el nou procés a executar.

  c) Només és útil per controlar temes relacionats amb la seguretat del nou procés a executar.

  d) El sistema del fork i exec va ser dissenyat a l'any 1973. Actualment es podria canviar per un sistema molt més potent i útil.

Pregunta 2

- Un proceso padre puede controlar el context de su proceso hijo:
  - Privilegios (Seguridad)
  - Tiempo ejecución
  - Memória disponible
  - Descriptores de ficheros -> E/S
  - …
- A UNIX, la interfície de programación prácticament no ha cambiado desde sus comienzos (1973), y continua siento muy utilizada hoy en dia ya que es sencilla, potente y portable.
- Sistemas de 2 pasos (fork y exec)
  - `fork()` -> el hijo hereda el contexto del padre (privilegios, ficheros abiertos, …)
  - `exec()` -> nuevo contexto

Pregunta 3

- **Marca la resposta que correspon a una aplicació que, en executar-se, genera un procés.**

  a) Un fitxer amb un script bash.

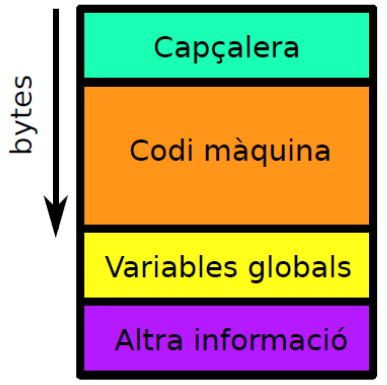
  b) Un fitxer amb un programa en C.

  c) Un fitxer amb un script Python.

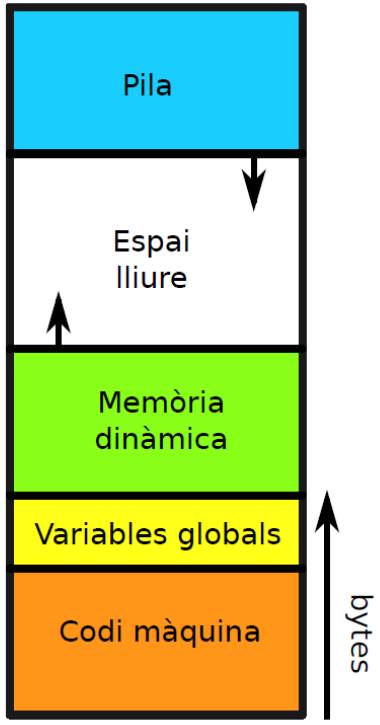
  d) L'intèrpret de comandes bash.

Pregunta 3

- **Programa:** Debe contener código máquina ejecutable per la CPU
- **Proceso**: Programa en ejecución.



Estructura d'un programa



Estructura d'un procés a memòria

Pregunta 4

- **La canonada és un sistema de comunicació interprocés que...**

a) … permet comunicar múltiples processos entre sí encara que no que tinguin una relació pare-fill.

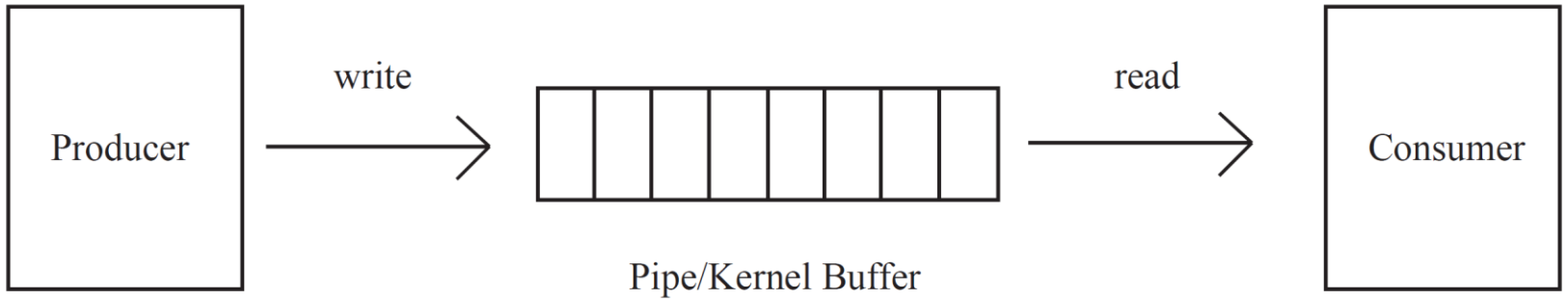
b) … únicament permet comunicar la sortida estàndard d'un procés amb l'entrada estàndard d'un altre procés.

c) … permet comunicar entre sí processos que tinguin una relació pare-fill.

d) … permet comunicar entre sí dos processos encara que no que tinguin una relació pare-fill.

Pregunta 4

- Una tubería és una forma básica de comunicación entre procesos
- Una tubería UNIX es un búfer del núcleo (kernel) con dos descriptores de archivo (fd), uno para escribir (para poner datos en la tubería) y otro para leer (para extraer datos de la tubería).



Anderson and Dahlin (2012). Operating Systems: Principles and Practice.

Pregunta 5

- **Les canonades fan servir un buffer intern del sistema operatiu que gestiona la comunicació interprocés. Què passa quan aquest buffer es troba ple?**

  a) Aquest buffer té una mida dinàmica (blocs de 64KB) i no s'omple mai.

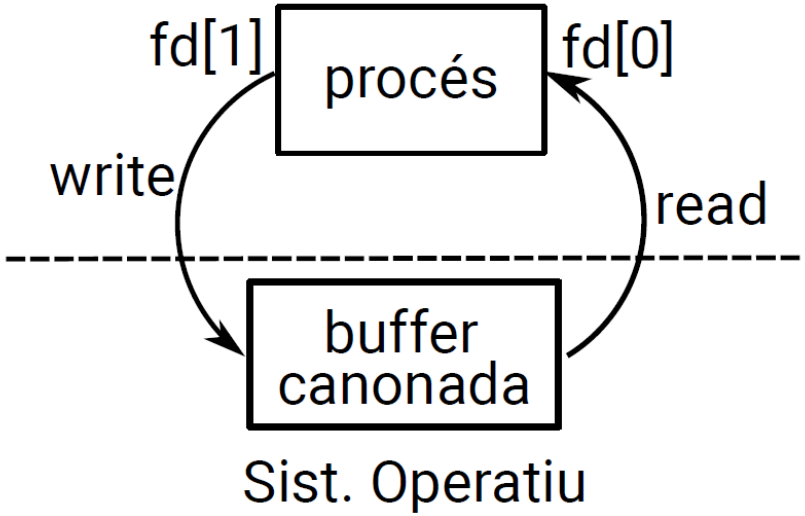
  b) Es produeix una interrupció i el procés fill s'atura.

  c) El procés que fa la crida read espera a que el buffer estigui buit.

  d) El procés que fa la crida write espera a que el buffer comenci a buidar-se.

Pregunta 5

- El buffer es interno del sistema operativo y es relativamente pequeño (64 KB)
- Si al leer el buffer éste esta vacío, el proceso que realiza la llamada a sistemas (read) se bloquea hasta que hayan datos disponibles.
- Si al intentar escribir el buffer éste se encuentra lleno, el proceso que realiza la llamada a sistemas (write) se bloquea hasta que el otro proceso comienza a vaciar datos.



Resumen

- Procesos y "*process control block*"
- Llamadas a sistemas -> comunicación de procesos entre sí a través de interfícies de programación (API)
- UNIX -> fork() y exec()
- Gestión de contextos
  - Redirección estándard a fichero (E/S) -> descriptores de fichero (dispositivos)
    - 0-keyboard
    - 1-stdout
    - 2-stderr
  - Tuberias/Canonades/Pipes

Ejercicio

- Buscar más información sobre los siguentes commandos/llamadas a sistema UNIX:
  - strace
  - dup, dup2
  - pipe

Los procesos

Indica la idea/concepto más importante que hayas aprendido

Mentimeter

Los procesos

Indica la idea/concepto menos claro

Gracias