Sistemas Operativos I 2023
Trabajo Práctico Obligatorio 3

Objetivos

- Analizar diferentes mecanismos de **comunicación entre procesos**.

Referencias

[1] Tanenbaum, Bos – Modern Operating Systems - Prentice Hall; 4 edition (March 10, 2014) - ISBN-10: 013359162X
[2] Douglas Comer - Operating System Design - The Xinu Approach. CRC Press, 2015. ISBN : 9781498712439
[3] Silberschatz, Galvin, Gagne - Operating Systems Concepts - John Wiley & Sons; 10 edition (2018) – ISBN 978-1-119-32091-3

Software y Hardware

Linux y Xinu. La versión de Xinu que utilizamos es para arquitectura PC (x86). Ejecutamos el sistema operativo Xinu en una máquina virtual llamada QEMU, que emula una PC básica.
El trabajo puede realizarse sobre las máquinas de los laboratorios (RECOMENDADO).
Quienes tengan Linux en sus casas, podrían intentar instalar todo lo necesario y llevarlo a cabo ahí también. Una tercera posibilidad es el acceso remoto RDP comentado en la web de la materia.

Ejercicio 1. *Implementar un programa en Linux que utilice* ***memoria compartida (comunicación entre procesos).***

a. Desarrollar un programa A que solicite al sistema operativo Linux una región de memoria compartida. Luego, el proceso A debe abrir el archivo
/usr/share/doc/util-linux/source-code-management.txt
(archivo de texto) , leer su contenido, y colocarlo en la región de memoria compartida creada.

b. Desarrollar un segundo programa, B, que le solicite a Linux el acceso a la memoria compartida, y presente en pantalla el contenido de esa región (la cual será el contenido del archivo de texto).

c. Utilice las funciones open(), read(), close() en a. para leer el contenido del archivo.

Documentación:
man 2 open
man 2 read
man 2 close
man 3 shm_open
man 3 shm_unlink

Ejercicio 2. *Implementar un programa en Xinu que utilice* ***pasaje de mensajes (comunicación entre procesos).***

Usted ha sido contratado por Shigeru Miyamoto para implementar en Xinu un video juego.
Él le explica que ha portado una versión de Galaga de la consola Game Boy Advance (GBA) a Xinu, y le solicita que lo termine.
Luego de una revisión usted detecta que el juego está lleno de bugs y hacks.
La paleta de colores es incorrecta, los disparos no siempre alcanzan al objetivo correctamente (detector de colisiones defectuoso), no existe puntuación, el código es horrible.

a. Divida el videojuego en al menos 3 procesos:
   - Un proceso 1 es el actual juego galaga.
   - Un proceso 2 mantiene las vidas y el puntaje.
   - Un proceso 3 control.

b. El proceso 3 control crea los otros dos procesos, y se queda esperando por un mensaje de finalización.
c. El proceso 2 debe mostrar en la pantalla (sobre el tapiz amarillo de fondo de la interfaz grafica de Xinu las vidas que le queda al jugador, y el puntaje. Este proceso se queda esperando mensajes.
d. El proceso 1 es el juego actual. Este proceso utilizará comunicación inter-procesos de Xinu, enviando mensajes:
   i. Cuando el proceso 1 detecta que la nave player es alcanzada por una nave enemiga (colisión) el proceso 1 le envía un mensaje al proceso 2 para indicar que el player perdió una vida.
   ii. Cuando el proceso 1 detecta que un disparo colisionó con una nave enemiga entonces el proceso 1 le envía un mensaje al proceso 2 para que aumente el puntaje de player.
   iii. Cuando el proceso 1 detecta que el jugador quiere terminar (por ej. el jugador presiona cierta tecla -elija UD. la tecla: observe como es el sistema de pulsaciones el teclado en el juego-) entonces el proceso 1 le envía al proceso 3 un mensaje de que el juego terminó y que el jugador quiere salir del juego.

e. Si el proceso 3 control obtiene un mensaje de salir debe finalizar todo el juego (todos los procesos relacionados con el mismo).
f. Bonus (optativo): agregar si el jugador perdió, ganó, matar bugs. Cualquier otra característica que ponga feliz a Shigeru Miyamoto.