

Sistemas Operativos

2012 Obligatorio 2

Objetivo

Familiarizarse y afianzar conceptos de programación concurrente.

Se pide

Se debe escribir en C un programa que implemente una simple aplicación de “chat”. El programa permitirá a múltiples usuarios conectarse a una sesión de chat en vivo. Los usuarios ingresarán mensajes de chat y sus mensajes se mostrarán en la pantalla de los demás usuarios. Los usuarios podrán elegir su nombre de usuario antes del comienzo de una sesión.

Una sesión de ejemplo se muestra a continuación:

```
./chat
Bienvenido.
Ingrese su usuario: Matias
(Ctrl-C presionado)
=>> Hola, Jorge! Como andas?
Matias: Hola, Jorge! Como andas?
(Ctrl-C presionado)
=>> Yo por comer algo.
Jorge: Mas o menos, toda la noche programando.
Jorge: Voy a ver si duermo un rato.
Matias: Yo por comer algo.
(Ctrl-C pressed)
=>> Si, esa tarea de SO está imposible.
(Ctrl-C pressed)
=>> salir
Chau.
```

Se deberá utilizar memoria compartida para la comunicación interprocesos. Para la sincronización de la comunicación se utilizarán semáforos.

Para la implementación se deben usar las bibliotecas *ipc* de C.

```
<unistd.h>
<sys/types.h>
<sys/wait.h>
<sys/stat.h>
<sys/ipc.h>
<sys/shm.h>
```

Se recomienda estudiar estas bibliotecas. Para facilitar también se dispondrá de una implementación más sencilla del manejo de memoria compartida y semáforos.

<shmem.h>

<semaforos.h>

Se dispondrá también de [semaforos.c](#) y [shmem.c](#) para que se pueda estudiar su contenido.

Cada proceso de chat básicamente lee de la memoria compartida y despliega los nuevos mensajes en la pantalla. Cuando un usuario desea escribir un mensaje, se presiona Ctrl-C y se ingresa el mensaje. El proceso entonces escribe el mensaje en memoria compartida y vuelve a leer y desplegar mensajes.

Una posible solución consiste en una porción de memoria compartida, donde de alguna manera se ordenan y almacenan los mensajes de cada proceso. Tener en cuenta que quizás se necesite otra porción de memoria compartida para albergar el contador de cantidad de procesos que están realizando lecturas sobre los mensajes que se encuentran en memoria compartida.

El primer proceso que comienza crea las memorias compartidas y los semáforos, y los procesos siguientes se asocian a esas memorias compartidas y semáforos.

Un pseudo-código para esta solución:

```
begin
    obtener memoria compartida
    si hay error
        crear memoria compartida
        crear semáforo
    sino
        obtener semáforo
    configurar manejador de señal para escribir mensaje
    while(1) {
        leer de la memoria compartida
        desplegar mensajes nuevos
        sleep x segundos
    }
end
manejador de señal
```

```
    obtener mensaje entrada estándar  
  
    si la entrada es "salir"  
  
        salir  
  
    escribir entrada en memoria compartida  
  
end
```

Consideraciones Generales

- Se proporciona un ejemplo de utilización de semáforos y memoria compartida.
- Se deberá escribir un breve informe explicando la solución implementada. Debe incluir una descripción del diseño del sistema, detalles de la implementación, dificultades encontradas durante el desarrollo y conclusión.
- Los programas deberán atender las señales del sistema, en particular CTRL + C (SIGINT) para escribir mensajes.
- Se recomienda leer los siguientes links sobre señales.
 - <http://www.chuidiang.com/clinix/senhales/senhales.php>
 - http://es.wikipedia.org/wiki/Se%C3%B1al_%28inform%C3%A1tica%29
 - <http://www2.dis.ulpgc.es/~itis-ps/signal/index.html>
- Se recomienda también familiarizarse con el comando ipcs de UNIX para ver los semáforos y segmentos de memoria compartidos, pues si no son destruidos “sobreviven” a la terminación del programa.
- Se debe entregar un makefile que compile todo el proyecto.
- Se publica un archivo llamado ejemploipc.tar.gz, el mismo contiene un ejemplo con el uso de las bibliotecas y manejo de señales, el mismo puede ser utilizado como base del proyecto.

Descripción de la entrega

- El trabajo se realizará en grupos de 3 o 4 integrantes.
- Se deberá entregar un correo electrónico al docente del curso.
- El correo deberá tener el siguiente formato:
 - **Subject:** so2012e3-1234567-1234568 (Donde 1234567, 1234568 son las cédulas de los integrantes del grupo sin dígito de verificación).
 - **Body:**
1234567 Nombre1 APELLIDO1
1234568 Nombre2 APELLIDO2
(Corresponden a los datos de cada integrante del grupo)
- El correo electrónico deberá tener adjunto un archivo llamado tarea2.tar.gz. (deberá contener

todos los archivos necesarios para la solución, así como un makefile para compilar el proyecto, y opcionalmente un archivo leame.txt con aclaraciones sobre la solución).

- El correo electrónico deberá ser de texto sin formato.

Fecha de entrega

La fecha límite para realizar la entrega es el domingo 18 de noviembre a las 23:59

Observaciones

No se aceptará bajo ninguna circunstancia una entrega realizada pasada la fecha límite estipulada. En caso de que una entrega no cumpla los criterios pautados en cuanto a los formatos y mecanismos establecidos, la misma puede no ser considerada como tal y descartada.