PRÁCTICA II. COMPUTADORES 2013/14

Ingeniero Industrial.

Escuela de Ingenierías Industriales. Universidad de Valladolid.

OBJETIVO

En esta práctica se implantará un árbol binario de búsqueda con objeto de que el alumno se familiarice con este tipo de estructuras de datos.

ESQUEMA DE TRABAJO

El trabajo, a realizar individualmente, consiste en:

1. Estudio bibliográfico previo.
2. Diseño e implantación de un programa en C.
3. Escritura de un breve informe.
4. Presentación del programa e informe y discusión con el profesor.

INFORMES

El informe escrito tendrá una extensión máxima de 10 caras (sin contar el anexo) y constará de:

1. Título, autor del trabajo y número de grupo de laboratorio.
2. Índice.
3. Descripción del problema y de las soluciones adoptadas, con especial hincapié en su originalidad y su robustez (mejoras sobre las funcionalidades exigidas en la práctica, tratamiento de errores,etc).
4. Resultados y conclusiones.
5. Bibliografía.
6. Anexo : códigos fuente. (¡Deben estar adecuadamente comentados!)

EVALUACIÓN

Tras la entrega del informe, el profesor procederá a la revisión del trabajo en presencia de del autor o autores (dos como mucho).

La calidad del trabajo será valorada de acuerdo con los siguientes criterios:

- Funcionamiento general de la aplicación, originalidad y robustez (mejoras, tratamiento de errores y casos excepcionales). *50% de la nota.*
- Estructuración y legibilidad del código: uso adecuado de ficheros cabecera, funciones menores de 30-40 líneas, comentarios, pocas variables globales etc. *25% de la nota.*
- Calidad del informe: presentación, claridad, facilidad de lectura*. 25% de la nota.*

DESCRIPCIÓN DE LA PRÁCTICA

1- INTRODUCCIÓN

La presente práctica servirá al alumno para conocer el paradigma cliente-servidor, la sincronización y comunicación de procesos concurrentes, y las herramientas de sincronización y comunicación que ofrece Unix.

A continuación se presenta una breve introducción al paradigma cliente-servidor (sección 2), la descripción de la práctica (sección 3), las mejoras opcionales (sección 4) y fecha de entrega (sección 5).

2- EL PARADIGMA CLIENTE-SERVIDOR

Un proceso puede proporcionar servicios a otros, tales como imprimir un documento, realizar cálculos, enviar cierta información, etc. En el modelo cliente/servidor, cuando un proceso desea un servicio que proporciona otro proceso, el primero le envía una cierta **petición**. El proceso que formula dicha petición se denomina **cliente** y el proceso encargado de atenderla se denomina **servidor**.

Los procesos clientes y servidores han de seguir un cierto **protocolo** de comunicación que define:

(a) Cómo se codifican las peticiones de los clientes y las respuestas del servidor: cómo se codifica cada petición concreta, en qué orden van los posibles parámetros, cuantos *bytes* ocupan, etc.
(b) Cómo se sincronizan los procesos: si el cliente puede seguir adelante tras enviar la petición (comunicación no bloqueante) o bien si debe permanecer bloqueado hasta que se reciba la respuesta del servidor (comunicación bloqueante).

Además el diálogo cliente/servidor suele ser bidireccional: el cliente envía información al servidor (tipo de servicio solicitado y parámetros correspondientes) y el servidor envía información al cliente (servicio finalizado, resultado del mismo, códigos de error en su caso, etc.).

3- DESCRIPCIÓN DE LA PRÁCTICA

En esta práctica se implantará un sistema cliente-servidor integrado por los siguientes elementos (*véase* la figura 1):

- Un proceso **servidor** que gestiona un árbol de búsqueda binario (con el programa realizado en la práctica 1) en base a las peticiones que le realizan los clientes. El servidor mantiene una lista con los MAX_CLIENTES procesos **cliente** activos (MAX_ CLIENTES es consignado por el usuario mediante la línea de comandos, al lanzar el proceso servidor.) Este proceso servidor (i) detecta los clientes que se incorporan al sistema, a los que identifica por su pid; (ii) queda a la espera de recibir las ordenes que envían los clientes; (iii) atiende las peticiones de los clientes y les devuelve confirmación. El proceso servidor termina cuando se pulsa un Ctrl-C.

- Una serie de procesos **cliente** idénticos, cada uno de los cuales solicita el alta al proceso servidor nada más ser lanzado. Cuando este le confirma el alta, presenta un menú con las siguientes opciones:

1. Introducir nuevo dato en el árbol
2. Borrar dato del árbol
3. Buscar dato en el árbol
4. Terminar

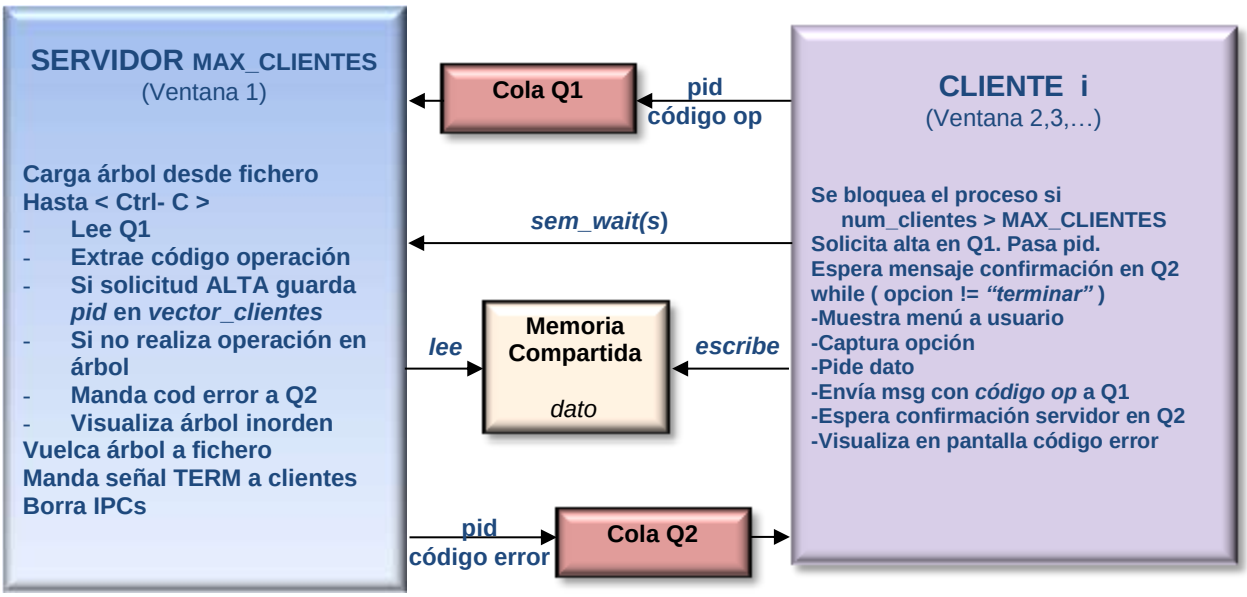

Figura 1

En las subsecciones siguientes se describen con mayor detalle los distintos procesos.

3.1- SERVIDOR

El proceso servidor puede operar como se propone a continuación (el pseudocódigo que se proporciona es simplemente una guía y no debe ser transcrito literalmente sino adaptado a la naturaleza y al funcionamiento de las llamadas al sistema que ofrece UNIX).

- *Crea colas Q1, Q2*
- *Crea memoria compartida*
- *Crea e inicializa semáforo mutex=1*
- *Crea e inicializa semáforo: s = MAX_CLIENTES.*

  // Reserva memo para guardar los pids en un vector.
- *vector_clientes = malloc(MAX_ CLIENTES * sizeof (pid_t))*
- *while(1)*
  - *msgrcv(Q1,pid,cod_op) // cliente ha solicitado realizar la operación cod_op*
  - *Si cod_op==ALTA*
    - *lee pid del nuevo cliente y lo anota en vector_clientes*
    - *num_clientes ++*
  - *Si cod_op==INSERTAR || cod_op==BORRAR || cod_op==BUSCAR*
    - *Lee info memoria compartida*
    - *Realiza la operación sobre inventario (si se puede)*
  - *msgsnd(Q2, pid, cod_error) // Envía a cliente pid confirmación operación*

- *Si en cualquier momento se recibe señal SOLICITUD_TERMINACION*

- *Borra ipcs*
- *Termina procesos cliente*

3.2- CLIENTES

Cada proceso cliente será lanzado en una ventana distinta a la del proceso servidor. Cada cliente solicitará al usuario la opción que desea llevar a cabo sobre el inventario. Esta se enviará al servidor y el cliente la confirmación o código de error. Este proceso continuará hasta que el usuario elija la opción finalizar.

Los procesos cliente pueden operar, a grandes rasgos, como se propone a continuación:

- *Obtiene identificador de colas, shm y semáforos s y mutex*
- *Si alguno no existe, termina.*
- *sem_wait (s)* *// se bloquea si hay más de MAX_CLIENTES*
- *msgsnd (Q1, pid, ALTA)* *// Solicitud de alta*
- *msgrcv(Q2,ACKN)* *// Espera confirmación de alta por Q2*

- *while opción!=terminar*
  - *muestra menu*
  - *scanf(operacion)* *//pide cadena por teclado al usuario*
  - *scanf(dato)* *//pide dato al usuario*
  - *sem_wait(mutex);*
  - *memcpy (shmemo,dato);//copia dato en memo compartida*
  - *msgsnd(Q1, pid,op)* *//envía msg con código operación a Q1*
  - *msgrcv(Q2, pid)* *//lee código error en mensaje con tipo pid*
  - *sem_post(mutex);*
  - *printf(código error);*

- *sem_post(S);* *//sube el semáforo para dejar paso a otro cliente*

IMPORTANTE: Los IPCs no se eliminan automáticamente cuando termina el proceso que los ha creado. Por ello, durante la programación y puesta a punto de aplicaciones que utilizan estos mecanismos es imprescindible listar regularmente los ipcs existentes en el sistema (mediante el comando **ipcs**) y eliminar los no deseados (mediante el comando **ipcrm**). Los semáforos POSIX no aparecerán en el listado que genera ipcs. Para visualizarlos basta con listar los ficheros del directorio */dev/shm* y su borrado se hace como en el caso de un fichero, con **rm**.

4. MEJORAS OPCIONALES

El programa propuesto es susceptible de numerosas mejoras que el alumno puede introducir libremente. Entre estas mejoras estaría la de comprobar periódicamente que ningún cliente ha caído. El bloqueo o la terminación anormal de un cliente impedirían subir el semáforo S, evitando el paso de un nuevo cliente.

Para esta gestionar el control de los clientes se podría incorporar el siguiente código al servidor:

- *Para cada cliente de vector_clientes*
  - *kill(pid_cliente, SOLIC_ACTIVO)*
  - *alarm(TIMEOUT)*

- *msgrcv(Q, WAIT)*
  (Si el mensaje recibido no es de pid_cliente esperar hasta que lo sea)

  (- Si mensaje no recibido al cabo de TIMEOUT:)
  (- Terminar pid_cliente)
  (- num_clientes --)
  (- sem_post(S))

Por su parte, los clientes, responderían a la petición de actividad del servidor escribiendo en la cola Q.

- *Si en cualquier momento se recibe señal de SOLIC_ACTIVO*
  - *msgsnd (Q, pid, ACTIVO)*

5. PLAZO DE ENTREGA

La **fecha de tope de entrega será el viernes 20.12.2012 a las 13:00 horas**

La fecha para la revisión y evaluación con el profesor será comunicada oportunamente a cada grupo. Es imprescindible que a la evaluación del trabajo de cada grupo acudan todos sus integrantes.