dit
UPM

Universidad Politécnica de Madrid
E.T.S.I. Telecomunicación
Dep. Ingeniería de Sistemas Telemáticos

EXAMEN EXTRAORDINARIO DE SISTEMAS OPERATIVOS
Convocatoria de Febrero. 5–2–97
PRIMERA PARTE (SIN LIBROS)

DURACIÓN: 1 hora y cuarto

Puntuación y longitud *máxima* —puede de hecho ser más corta— de la respuesta, en cada pregunta. **Una página es una cara**.

Primera pregunta

El direccionamiento en los computadores modernos está evolucionando a direcciones formadas por 64 bits. Analizar las posibles influencias de esta evolución sobre el esquema de direccionamiento virtual en base a regiones, así como sobre el trabajo que hacen los enlazadores (*linkers*).

(*3/4 página; 2 puntos*)

La razón fundamental para ir a direcciones de 64 bits es que la posibilidad económica de comprar cantidades de memoria central de más de 4 millardos de octetos ya ha dejado de ser una ficción.

Con independencia de lo anterior, el tamaño de un espacio virtual de direcciones de 64 bits es tan extraordinariamente amplio, que es un reto para la imaginación sólo pensar en las posibles aplicaciones que pueda tener.

*Desde el punto de vista de los enlazadores (*linker*) clásicos, éstos pueden verse como "reutilizadores" del espacio de direcciones lógicas. Con 64 bits de direcciones, podría entonces pensarse en que esta reutilización del espacio lógico de direcciones fuera innecesaria, teniendo al compilador de una máquina asignando direcciones virtuales siempre diferentes, ¡y sin que por ello se le acaben éstas en toda su vida! …Esa posibilidad, sin embargo, no parece que pudiera dar buenos resultados si se tiene en cuenta que normalmente se desea poder trasladar código ya compilado de un computador a otro, sin que sea necesario recompilarlo.*

En cuanto al espacio virtual de direcciones en base a regiones, los 64 bits de dirección hacen posible un número prácticamente ilimitado de las mismas (de tamaño también prácticamente ilimitado), lo cuál permite en especial tener tantas regiones para bibliotecas de enlace dinámico como se quiera, así como procesos con un número ilimitado de (pilas para) hebras. Y, algo quizás de la máxima importancia, permite pensar en la memoria virtual de un proceso como un conjunto de regiones que pueden estar distribuidas a lo largo de los nodos de una red de un tamaño hasta ahora inimaginable (varios millardos de nodos, por ejemplo).

Segunda pregunta

¿Qué significa que una operación de un sistema de ficheros (u otro servidor) remoto sea **idempotente**? ¿Qué utilidad tiene? ¿Cómo se hace uso de esta característica?

(*1/2 página; 1 punto*)

Se dice de una operación que es idempotente, si el efecto de realizarla más de una vez (completa o parcialmente, pero completamente al menos la última vez) en secuencia es equivalente a realizarla una sóla vez.

Las operaciones idempotentes permiten utilizar un mecanismo muy simple, a base de reintentos, para la recuperación de fallos temporales en la interacción entre cliente y servidor, debidos, bien a la pérdida de mensajes de petición o de respuesta, o bien a la caída temporal (con posterior recuperación) del servidor.

Tercera pregunta

Explicar con ayuda de un esquema el funcionamiento del algoritmo de Cristian de sincronización de relojes.

(*1/2 página; 1 punto*)

En el algoritmo de Cristian, un proceso cliente que desee sincronizar su hora con otro proceso servidor (de hora), le envía a éste una petición a la cuál el servidor responde, nada más recibirla, con su hora actual.

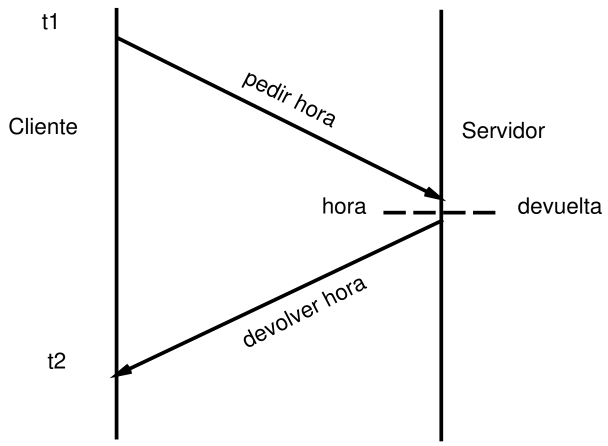

Figura 1: Algoritmo de Cristian para sincronización de relojes

El cliente entonces ajusta su reloj propio a la hora recibida del servidor, más el tiempo de propagación del mensaje recibido. Este último tiempo lo estima como la mitad del tiempo total de "ida y vuelta" (desde que envió la petición hasta que recibió la respuesta), que puede medir con exactitud con su propio reloj. Esta estimación se basa en suponer idénticos los tiempos de transmisión de los mesajes de ida y de vuelta, lo cuál significa, en todo caso, una cota de error máximo en valor absoluto igual, de nuevo, a la mitad del tiempo de ida y vuelta, pero restando ahora el tiempo mínimo de transmisión de un mensaje por la red.

Cuarta pregunta

Explicar con ayuda de un gráfico un algoritmo que realice multienvío con orden total (no necesariamente causal)

(*1/2 página; 1 punto*)

Para realizar multienvío con orden total, se puede tener un proceso secuenciador que recibe en primera instancia todos los mensajes y los reenvia después de haberles asignado un número consecutivo de orden.

Este algoritmo, sin embargo, no es robusto. Uno que sí lo es, consiste en que los procesos disponen de un contador creciente, cuyo valor incrementan en una unidad y devuelven al emisor

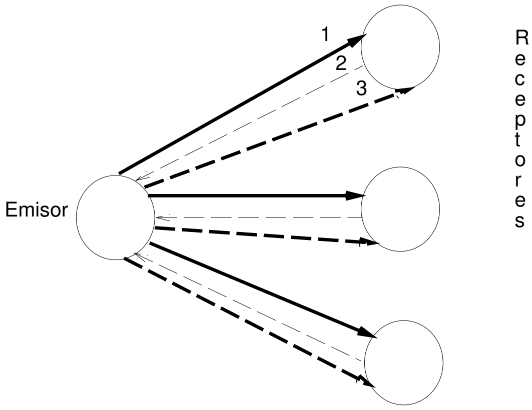

Figura 2: Multienvío con orden total

de todo mensaje que reciban, como posible valor de orden a darle. El emisor elige entonces el mayor de los valores así recibidos (rompiendo posibles empates con los números de identificación de los procesos receptores) como número de orden del mensaje correspondiente, y se lo comunica de vuelta a todos los receptores. Estos, aparte de apuntarse el número de orden decidido para el mensaje, toman este valor como el nuevo de su contador, si el actual es menor.

Quinta pregunta

Dar un ejemplo sencillo de dos transacciones que se entrelazan y acceden a dos variables comunes, y cuyo efecto global es, sin embargo, *serialmente equivalente*.

(*1/4 página; 1 punto*)

Variables x e y, L=lectura, E=escritura, transacciones 1 y 2:

L1(x); E1(x); L2(x); L1(y); E2(x); E1(y); L2(y); E2(y)

dit
UPM

Universidad Politécnica de Madrid
E.T.S.I. Telecomunicación
Dep. Ingeniería de Sistemas Telemáticos

EXAMEN EXTRAORDINARIO DE SISTEMAS OPERATIVOS
Convocatoria de Febrero. 5–2–97
SEGUNDA PARTE (CON LIBROS)

DURACIÓN: 1 hora y cuarto

Puntuación y longitud *máxima* —puede de hecho ser más corta— de la respuesta, en cada pregunta. **Una página es una cara**.

Primera pregunta

El sistema de supervisión y extinción de incendios de una nave industrial está formado por tres procesos, **A**, **B** y **C** que, para intentar que el sistema sea robusto, ejecutan sobre computadores independientes comunicados por una red fiable. El proceso **A** se encarga de detectar posibles incendios, el **B** de abrir las válvulas de los aspersores que llenan la planta de anhídrido carbónico para apagar el fuego, y el **C** de inundar la planta con agua, como solución "definitiva" en el caso de que **B** falle.

En cuanto detecta un fuego, el proceso **A** envía un mensaje de difusión (*broadcast*) a los otros dos. El proceso **B**, nada más recibir ese mensaje, abre los aspersores y envía a su vez otro mensaje al proceso **C**, para notificarle que los ha abierto. Por último, y para tener un nivel adicional de seguridad sin causar daños innecesarios, el proceso **C** sólo suelta el agua que inunda la nave si, transcurrido un cierto tiempo desde que **A** le ha notificado un fuego, no ha recibido del proceso **B** el correspondiente mensaje de confirmación de apertura de aspersores.

En un momento determinado se produce un fuego, que el proceso **A** detecta y el sistema en su conjunto gestiona satisfactoriamente (véase la figura). Sin embargo, justo a continuación se detecta otro fuego, y esta vez el sistema responde erróneamente, con mensajes intercambiados en un orden incorrecto, con el resultado de que se produce una catástrofe innecesaria —el proceso **C** inunda la nave de agua cuando ya los aspersores de gas carbónico habían ya entrado en funcionamiento para apagar el segundo fuego—.

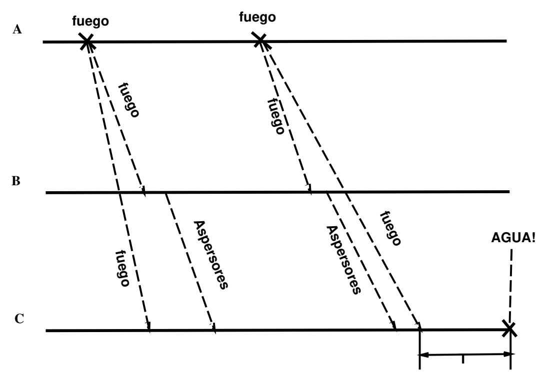

Figura 3: Sistema de gestión de incendios

Lo que ha ocurrido es que al llegar el segundo mensaje del proceso **B** al **C** en mal momento, el proceso **C** suelta el agua, creyendo que no ha habido respuesta del proceso **B** al segundo fuego.

1. ¿Qué (ausencia de) característica del sistema de comunicaciones ha causado la catástrofe?

(*una o dos líneas; 1/2 punto*)

2. ¿Qué técnica de comunicaciones debería haberse usado para evitarlo? ¿Por qué?

(*dos líneas; 1/2 punto*)

3. Cómo se podría implementar tal técnica en este ejemplo?

(*1/4 página; 1 punto*)

1. El fallo se ha producido porque el sistema de comunicaciones no respeta la causalidad (potencial), y por ello ha tardado demasiado en llegarle al tercer proceso la segunda notificación de fuego del primero.

2. La forma de corregir el problema anterior es haciendo que el sistema de mensajería respete el orden de causalidad (potencial).

3. Se puede implementar el orden causal en este sistema de mensajería haciendo simplemente que el tercer proceso se comporte, en lo que respecta a los mensajes de fuego y apagado enviados por los otros dos, como un **observador global**. *Suponiendo que existe un tiempo máximo de transmisión de mensajes en el sistema (lo cual es necesario en todo caso para poder garantizar que los fuegos se apagan a tiempo), basta con usar relojes lógicos de Lamport y que el tercer proceso espere, antes de aceptar cualquier mensaje, durante ese tiempo máximo de transmisión, por si le llega algún otro mensaje con hora lógica menor.*

Segunda pregunta

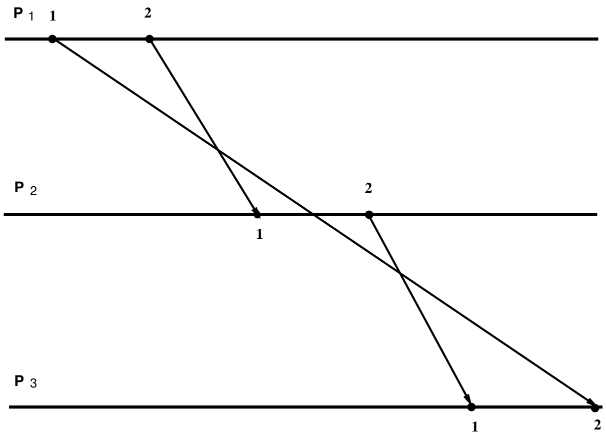

Figura 4: Computación distribuida

Para la computación distribuida que se muestra en la figura, se pide mostrar el retículo de estados globales coherentes.

(*1 página; 2 puntos*)

En este caso el retículo se convierte en una simple lista:

(0,0,0)—(1,0,0)—(2,0,0)—(2,1,0)—(2,2,0)—(2,2,1)—(2,2,2)