Clase 4-hilos I

miércoles, 9 de abril de 2025 17:17

Hilo:

- Hlo de control de una tarea secuencial que s va ejecutando en la cpu.
- Poner un proceso en ejeución seria poner un hilo.
- Proceso es la unidad de administración de recursos cpu, pero el que ejecuta en la cpu es un hilo.
- Proceso:
  - Programa en ejecución
  - Unidad básoca de administración de recursos.
  - No asigno hilo, sino proceso.
  - Hilo es la unidad de procesamiento eso si
- Proceso hace lo que quiere dentro del espacio de direcciones
- PCB tiene todo lo del proceso

Al principio deciamos uqe cada proceso tenia un solo hilo de ejecucioón. Si tenia que hacer una es o lo que sea lo sacaban y mientras entreba otro archvio a la cpu quedaba ocioso, no podia hacer nada. Se ejecutaban secuencialmente las instrucciones y si se bloqueba arrancaba después. Tenia que hacer context switch para que entre un proceso. Eso generaba mucho tiempo improductivo pq tenes que hacer más cosas para cambiar.
Evolucionó el hardware: podiamos empezar a poner dos procesadores, o más cores a un proce.
Dual processor: dos o más cpu en una maquina
Dual core: muchos nucleos en un proce
Hyper threading: processar muchos hilos en paralelo dentro de un mismo proce. Simula dos procesadores lógicos dentro de uno fisico

Hoy divido un proceos en distintas tares (me quedó una diapo acá)

Software también evolucionó: necesitamos poder correr varias tareas en simultáneo y evitar bloquearse.
Desde el desarrollo del software se empezaron a crear librerias para separar hilos
Es convenienvte tener varios hilos de cotrol el el mismo espacio de direcciones. Ejecutarlos casi en paralelo, como si furan separados
Los hilos comparten el espacio de direccioes, no tengo que salir para acceder a datos.

Unidad básica de utilización de cpu:hilo
Unidad básica de asignación de recursos: proceso

Proceso es un conenedor con espacio de direcciones, unidad de propiedad de recursos y threads
Hilo: unidad de trabajo, hilo que está pueto actualmente en el procesador.
Por qué hilos?
Porque quiero poder hacer más de una cosa a la vez
Con n cpus pueden ejecutarse n threads al mismo tiempo
No todo el software aprovecha sistema mutihilos
No todo software aprochevha la arquitectura multihilos

Ventjaas de hilos:

- Sincronización de procesos: usando un solo hilo, el sistema operativo se ocupa de todo. Trata de hacerlo justo, pero a veces no conviene. Usar hilos da la liertad de que sea el programador quien decide cuándo le tca a cada proceso
- Economía:qué referencia un proceso' pcb. Cuando creo un hilo, no. Es más barato crear un hilo que un proceso, usa una estructura más liviana.

System call. Bloqueante-> usar hilos me auda a que no se bloquee tanto y aprovechar mejor mi quantum
Proceso puede aprovechar mejor el quantum que le toca

Cada hilo tiene:

- Estado de ejecucíon
- Contexto de un procesador
- Stacks (usuario, kernel)
- Acceso a memoria y recursos del proceso(datos que comparten todos los hilos)
- TCB-----> thread control block, tlb de hilos

Los hilos tienen estados (parecido a procesos.) ready, bloqueado, nuevo, etc

Problemas y soluciones:

- Con procesos el context switch es un garron
- Con hilos, con vaambiar los registros es muhco m(as facil, no se tiene que involucrar el so.

- Crear un hilo es algo que se hace denro de, proceso, casi como definir una variable
- Crear un proceso es crear pcb, lo hace el so

Hilo ult hilo a nivel de usuario no son visibles al so y se planifican dentro del proceso. Despues hay klt kernel lt y ahi si son hilos posta o algo

Iprotección:
En procesos, el so hacia que no te metieras en el espacio de direcciones de otro proceso
En hilos,la proteción la hace el programadror. Todos los hilos comparten el ismo espacio de direcciones. Un hilo podria bloquear la

ejecución de otros

El sleep llama una sys call que manda a dormir el proceso--- y con éll me saca los hilos. Es una pavada usarlos como en el ejemplo

Short u long time scheduler están pero como doble: en la librería de hilos y posta para procesos
User level thread:

ULT
Ventajas:
Aplicación en modo usuario se encarga de gestionr los hilos usando una biblioteca. Kernel no se entera de que existen estos hilos.
Intercambiar hilos ese facil porque comparten el espacio de direcciones.
Plaificacion independiente, cada proces los planifica como les conviene
Hay que reemplaizar las llamadas al sistema bloqueantes a otras que no bloqueen, tipo sleep, ojo
Postabilidad: pueden correr en distintas plataformas.

desventajas
~~Desventajas de ult: no puedo ejecuar hilos del mismo proceo en varios procesadores.~~
Si un hilo hace un lage fault, se bloquea todo
Un hilo puede monopolizar odo.
Se puede bloquear el proceso por una syscall bloqueante
Sirve hasta ahi

KLT: administración de hilos las hace el kernel.
Ventajas:
Puedo multiplexar hilos del mismo proceso en varios procesadores.
Desventaja:
Nvoluvro al kernel, tengo que cambiar el modo de ejecución

Se convinan klt y ult
La creación de hilos se hace a nivel de usuarios, se mapean a una cantidad igual o menor de klt.
Haciendo esto, un hilo puede bloquearse y el resto del mismo proceso sigan. Tengo una visivilidad que no tená

Modelo uno a uno: cada vez que creo una ult se mapea con un klt.
Cuando necesito un ult, se crea un klt
La concurrencia o paralelismo es mucha pq cada hilo puede correr en un proce
Muy costoso pq hay que crear tantos klt como ults.

Modelo muchos a uno
Muchos hilos de usuario se mapean a un klt.
Si un hilo se bloquea, todos se bloquean.
Soluciona temas de portabilidad. Puedo laburar en sistemas sin hilos con hilos

Modelo muchos a muchos
Muchos ult mapean a muchos klt.
No lo entiendo creo, pero es como un punto medio