Procesos

Proceso: Unidad de actividad que se caracteriza por la ejcución de una secuencia de instrucciones (que contiene el micro), un estado actual, su memoria de trabajo y un conjunto de recursos del sistema asociados.

Cada proceso tiee su contador de programa, registros y variables asilados de otros procesos.

Un proceso se rige en pequeñas porciones conocidas como páginas (tabla de paginación).

El SO se encarga del IPC (Inter-Process Comuncation).

Crear un proceso

- Arranque del sistema
- Desde un proceso, por una llamada al sistema
- Petición deliverada del usuario
- Inicio trabajo por lotes

Terminación de un proceso

- Salida normal
- Salida por error
- Error fatal
- Eliminado por otro proceso

Estados

- Nuevo
- En ejecución
- Listo/preparado (Está esperando un intervalo de tiempo para que pueda usar el
- procesador)
- Bloqueado
- Terminado

Listo/preparado: Está esperando un intervalo de tiempo para que pueda usar el procesador

En ejecución: Está usando el procesador. En este estado el sistema operativo puede decir interrumpir el proceso o el mismo proceso puede bloquearse a sí mismo.

Bloqueado: El proceso se encuentra en este estdo si ha sido interrumpido por el OS o por sí mismo por la espera de uno evento (I/O).El proceso se va a la cola de lista de procesos.

Nota: Algunos autores muestran los estado “Nuevo” (proceso antes de “Listo”) y “Terminado” (No hay más ejecución ni espera del programa: “muere”).

Transiciones

- Nuevo a Listo

Nuevo a Listo: Cuando el proceso es admitido a la cola de procesos.

Listo a Ejecución: El scheduler despacha el proceso.

Ejecución a Listo: El OS interrumpe el proceso cuado termina su intervalo de tiempo asinado.

Ejecución a Bloqueado: El proceso se bloquea cuando espera una E/S o algún evento.

Bloqueado a Listo: Termina operación de E/S o evento.

Ejecución a Terminado: Cuando el proceso termina.

Gestor de excepciones

- Catastróficos (De detiene el OS/deja de funcionar)
- No recuperables (Atrapa excepción y la maneja (ejemplo: división entre cero))
- Recuperables (Se puede modificar el código y se recupera el proceso involucrado)

Caraterísticas de los procesos

- Código, datos, memoria, dinámica (Heap) Pila (Stack)
- Procesos creados a partir de otros procesos (padre, hijo)
- El OS decide qué recursos comparte
- El OS decide qe pasa si el proceso Padre muere
- El OS genera un PCB (Process Control Block) de cada proceso nuevo (en la etapa “Nuevo” a “Listo” se crea el PCB y en la de “Ejecución” a “Terminado” muere el PCB)

.model (SMALL, TINY, LARGE) .stack 10 (por defecto 1KB) .data var1 DB, 10, 13, “Hola”, “$” (variables) Código va aquí .code Código va aquí END

10: Retorno de carro 12: Salto de línea $: Indicador de interrupción de terminación de impresión de datos

En los procesos comunes su código es inmutable.

PCB (Process Control Block)

ALgunos dependiendo del OS (del diseño) puede o no guardar algunas de las siguentes cosas en su estructura:

- Identificador de proceso (PID)
- Estado del proceso
- Contador del programa (PC)
- Valores de registros del CPU
- Espacio de direcciones de memoria
- Prioridad
- Lista de recursos asignados
- Estadísticas del proceso
- Datos del propietario
- Permisos

Listas

- Lista de procesos del sistea (job queue)
- Cola de procesos listos (ready queue)
- Cola de espera de dispositivos (device queue)

  RQ -> CPU —>————————>–>>—>—>_———-| |—-<—<—-(interrupción)-<–(ocurre interrupción)| |—-<—<—-(E/S resulve)-<–(device queue)-<-()| Missing info

Cada dispositivo tiene su propia device queue.

Cambio de contexto (Cambio de un proceso por otro)

- Salvar el “estado” del proceso a su PCB (cambiar estado). Su “estado” en este contexto se refiere al valor los registros, memoria, etc. Se guarda con el tipo de estado (Bloqueado o Listo)
- Seleccionar otro proceso o parte de un algoritmo equitativo
- Cargar el “estado” del proceso a ejecutar desde su PCB
- Ejecutar el nuevo proceso (cambiar su estado a en ejecución) (cambiar su estado a en ejecución)

Tarea

(Cualquier lenguaje de programación)

¿Cómo se crea un proceso? ¿Cómo se crea un hilo?

Thread (hilo o proceso ligero): Unidad básica de uso de CPU

- Tiene su propio uego de registros, pila, PC y prioridad (asignado por el SO o Usuario)
- Comparten código, datos y recursos
- Un hilo puede pertenecer a una sola tarea (proceso pesado)

Proceso pesado (foto de tabla de código, datos y recursos).

Implementación

- User Threads (hilos a nivel de usuario).
- Kernel Threads (Hilos a nivel de kernel). El scheduler ven todos los procesos como independientes

IPC (Inter Process Comunication). Controla a los hilos a nivel de kernel.

Ventaja

- Aprovechamiento de multiprocesador
- Comparten memoria y recursos
- Economía: Cambio de contexto más agíl
- Respuesta: Mejor tiempo de respuesta

Desventaja

- Complejidad de programación

Concurrencia

Concepto lógico de la existencia de varias actividades ejecutpandose simultáneamente, gracias a la habilidad de distintas partes de un programa de ser ejecutado en desorden o en orden parcial sin afctar el resultado final. (síncrono o asíncrono)

Algunos autores dicen que para que haya concurrencia los procesos debe compartir recursos o que son síncronos (que un proceso tenga que esperar a otro para que pueda realizar otra tarea).

Contextos:

- Varias aplicaciones
- Aplicaciones estructuradas (Una aplicación se divide en varias y converge)
- Estructura del sistema operativo (servicios (Windows), Daemons (Linux))

Modelos de computadora:

- Multiprogramación con un único procesador (No al mismo tiempo; "paralelos")
- Multiprocesador (varios procesos al mismo tiempo)
- Multicomputadora

Libro: Daemon de Daniel Suarez

Motivaciones:

- Compartir recursos lógicos (archivos/servicios)
- Compartir recursos físicos
- Acelerar cálculos (hilos y convergen)
- Modularidad (Una entidad gigante se puede dividir en partes pequeñas)
- Comodidad (Ya no se hace monotarea: nos permite ejecutar varias apps al mismo tiempo)

Problema de concurrencia:

- Carrera o competencia (compiten por recursos)
- Postergación o aplazamiento (se bloquean porque bandera no se desactiva)
- Espera circular (Se cicla; A requiere a B y B requiere a A)
- No apropiación (Un proceso no puede apropiarse de un recurso. Tiene que ver la razón por la cual no puede apropiarse del recurso)
- Espera ocupada (Un proceso espera un recurso en su tiempo porque otro proceso tiene ocupado un recurso aunque este esté suspendido)
- Ocupar y esperar (Ejemplo: Un proceso A ocupa recurso A y un recurso B, pero un proceso B está ocupando el recurso B, por lo tanto el recurso A no está siendo utilizado por nadie porque el proceso A lo apartó porque requiere el proceso B para poder continuar)

Mutex (Mutual Exclusion)

Evitar que entre más de un proceso a la vez a la sección crítica.

En microkernel: Una parte de un programa se ejecuta en modo usuario y otra en modo privilegiado; la parte ejecutada en motod privilegiado es la **sección crítica**.

Una llamada al sistema es cuando pedimos:

- Solicitar dispositivo
- Solicitar archivos
- Solicitar servicio/daemon
- Comunicación

Mecanismo de Mutex

- Candados o cierre de Mutex
- Algoritmo de Dekker
- Algoritmo de Peterson
- Semáforos
- Monitores
- Paso de mensajes

Tarea (11 de septiembre de 2019)

Con mi equipo

Tema: Algoritmo de Peterson

Paso de mensaje Mutex

Candados

Limita el acceso por hilo

Candados en DBs

Semáforos

Monitores

Son estructuras abstractas para se usadas por un hilo de ejecución.

(Parecidos a los semaforos pero mejor)

Componentes

- Inicialización
- Datos privados
- Métodos del monitor
- Cola de entrada

Ejemplo del baño

Tarea

Programa

Crear archivo vacío

Hilo por cada caracter de la cadena

Escribir en un archivo los caracteres

Manejar concurrencia con algún método visto en clase

Planificador (Scheduler)

Componente funcional muy importante de los sistemas operativos multitarea el cuya función consiste en repartir el tiempo disponible de un microprocesador entre todos los procesos que están disponibles para su ejecución.

Niveles

- Planificador a corto plazo (dispatcher): Aquel que asgina el pedacito de tiempo a los procesos
- Planificador a mediano plazo (swapping): Aquel que hace el intercambio entre la memorias
- Planificador a largo plazo: Aquel que crea y destruye PCBs

Objetivos

- Justicia: Asignar a los procesos un tiempo justo
- Optimización de recursos o de su acceso
- Máxima capacidad de distribución
- Seguridad de prioridades
- Predictivilidad: Predecible siguiente proceso a ejecutarse
- Minimizar sobrecarga (saturación de CPU)

Para manejar los procesos se basa en ciertos criterios.

Criterios

- Equidad
- Eficacia: Mientras más tiempo funcione el CPU, mejor
- Tiempo de respuesta: (tiene que ver con si es por lotes o interactivo) Debe se rápido para proceso de tipo interctivo
- Rendimiento: Mientras más proceso por hora, mejor
- Limitar peticiones E/S: Manejar si un proceso tarda mucho esperando dispositivos E/S
- Si es por lotes o interactivo

Tipos

- Expropiativos: Cuando asignamos tiempo a cada proceso
- No expropiativos: Ejecutamos un proceso y se espera a que termine para poder empezar otro

Técnicas

- A plazo fijo: Viene de la mano con no expropiativos. Tienen mucha prioridad y poca duración
- FIFO (First In First Out): No muy dependiente del plazo
- SJF (Short Job First)
- SRTF (Short Remaining Time First): El que tiene un menor tiempo de espera a ejecutarse el prioridad. Al que le queda poco por terminar
- HRN (Highest Response Ratio): "El más justo". Usa una fórmula $P = (w + t)3$
- RR (Round Robin): Cola para multiprocesos. Si no alcanza a terminar el proceso en un plazo predeterminado, entonce el proceso regresa la cola.
- Colas Multi-Nivel: Es como RR pero con 3 colas: alta prioridad, media prioridad, baja prioridad.

Un planificador puede utilizar técnicas mixtas, no solo una.