Resumen SO

Cristian Escudero

21 de abril de 2012

1. Tipos de SO

El **SO** es un **administrador** de recursos: procesos, memoria, e/s, comunicaciones, usuarios, seguridad.

1.1. Clasificación según su estructura

Monolíticos. Están constituídos por un conjunto de rutinas fuertemente entrelazadas. Aunque **eficientes**, son difíciles de modificar debido a su falta de flexibilidad.

Jerárquicos. Proponen niveles de construcción del SO, que van desde el *usuario* (en el más alto nivel), al *hardware* (en el más bajo nivel):

> **usuarios** → archivos → e/s → comunicaciones → memoria → μP → **hardware**.

El más conocido es el modelo **THE** (de Dijkstra). Otra ejemplificación jerárquica es la de *anillos*, similar al THE. Los algoritmos del sistema se construyen distribuídos en cuatro anillos: del 0 (para el SO) al 3 (para las aplicaciones).

Cliente-Servidor. Es el modelo más reciente. Las aplicaciones del SO pueden solicitar (**cliente**) u ofrecer (**servidor**) un procedimiento. Las aplicación puede entonces variar en cualquier momento entre ser *servidor* o *cliente*. El **núcleo** del sistema es el encargado de establecer todas las comunicaciones y pedidos

Máquina Virtual. Se da en sistemas multiprocesos, en el cual se le *"hace creer"* al proceso que dispone de la totalidad del HW y del SO.

1.2. Según sus servicios

- Por el nro. de usuarios:
  - Monousuario
  - Multiusuario
- Por el nro. de tareas:
  - Monotarea
  - Multitarea
- Por el nro. de procesos:
  - Monoprocesador
  - Multiprocesador
    - Asimétrico
    - Simétrico

2. Procesos

2.1. Administración de Procesos

Un **proceso** es un programa en ejecución con un entorno asociado. Está constituído por áreas de código y de datos. Esta última se divide en el área que usa efectivamente el proceso y el área de la pila.

Los SO actuales trabajan con múltiples procesos usando una técnica que se conoce como *time-sharing*, que distribuye el tiempo del μP.

El tiempo asignado para uso del μP se conoce como *quantum*, y la forma en la cual el sistema conoce el tiempo que va a pasando es a través de una interrupción externa: la **IRQ0** del reloj.

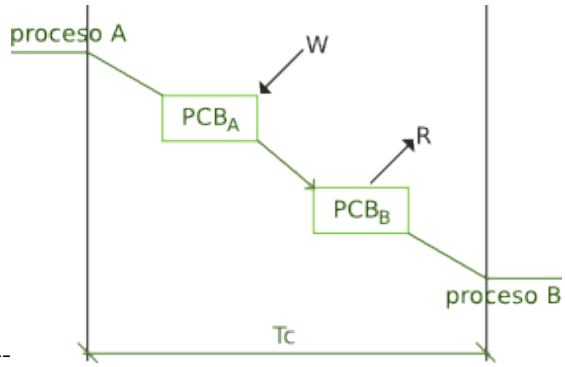
Al ser la IRQ0 esencial para el manejo de la multiprogramación, los programas no pueden enmascarar interrupciones. La instrucción para enmascarar, DI, es una instrucción propia del *modo protegido*.

2.2. Cambio de Contexto y PCB

Se produce un **cambio de contexto** cuando se pasa el control del procesador de un proceso a otro.

El **contexto** hace referencia al proceso y se expresa prácticamente en una estructura de almacenamiento conocida como **PCB** (bloque de control de proceso). El SO genera un PCB para cada proceso que se crea.

La conmutación entre un proceso y otro exige la actualización del PCB del proceso que se deja y la lectura del PCB del proceso que va a tomar al μP. Esta acción de cambio de contexto debe durar el **menor tiempo posible**, ya que es uno los indicadores del uso del procesador:



$$E = \frac{T_p}{T_c + T_p} \cdot 100\,\%$$

--

Normalmente el PCB contiene la siguiente información:

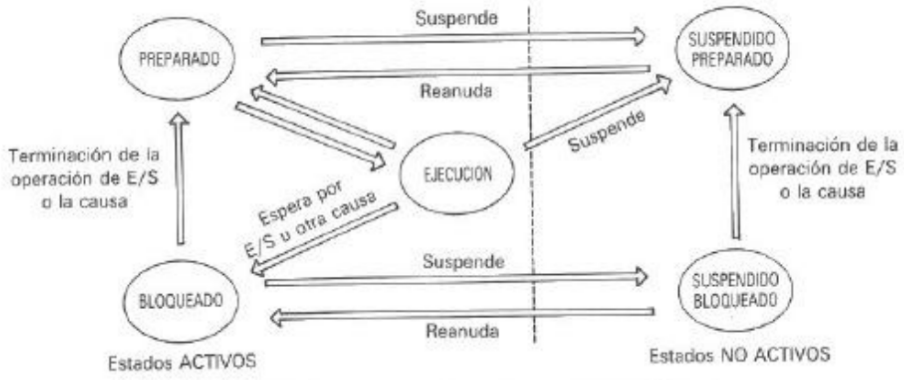
- Archivos/recursos en uso.
- Contenido de los registros internos del procesador.
- Ocupación de memoria externa/interna.
- Privilegios.

2.3. Estados de Proceso

Se pueden dividir en **activos** e **inactivos**. Entre los activos encontramos los estados de **ejecución**, **preparado** y **bloqueado**.

Los dos primeros están relacionados con el uso del procesador y es fácil reconocerlos en el *time-sharing*. Un proceso está *bloqueado* cuando en su ejecución se solicita un recurso que no está disponible.

Entre los estados inactivos encontramos los estados de **suspendidos** divididos en **bloqueados** y **preparados**.



Estados de un proceso y sus transiciones.

--

2.4. Operaciones sobre Procesos

La creación de procesos en cualquier sistema se realiza normalmente a partir de cuatro métodos:

1. Desde la **inicialización** sea porque se encuentra en el listado del sistema o en alguno de los registros de inicialización/configuración.
2. Porque lo ejecuta otro proceso.
3. Porque lo ejecuta el usuario, a través del *shell*.
4. Porque está incluído en un archivo por lotes.

La creación de los procesos puede ser **jerárquica** o **no jerárquica**. En el primero el hijo hereda parte del entorno del proceso padre, y si se elimina el proceso padre, al mismo tiempo se eliminan todos sus procesos hijos. En una creación *no jerárquica*, el proceso es independiente de su creador.

Todas las operaciones o acciones sobre los procesos se realizan o ejecutan a partir de llamadas al SO. Será un **fork** (unix/linux) o un **createprocess** (windows/api) para crear un proceso, un **kill** o **terminateprocess** para eliminarlo, o un **exit** o **exitprocess** para terminarlo.

2.5. Modificación de Privilegios

Los privilegios de un proceso pueden ser:

- **Estáticos.** No cambian durante toda la "*vida*"del proceso.
- **Dinámicos.** Pueden ser modificados a través del usuario o del SO.

Lo normal de los sistemas actuales es que el SO modifique los privilegios entre diferentes niveles. Un proceso normalmente se inicia con un nivel en particular y su valor se modifica por más o menos a partir de este.

2.6. Subprocesos

Son hijos de otro proceso, que heredan el entorno de éste pero que no lo necesitan cuando se ejecutan los cambios de contexto. De esta forma, la única diferencia entre "*hermanos*.[es] sólo el contenido de los registros internos del μP.

Para la administración de subprocesos el SO maneja una tabla de subprocesos.

2.7. Tipos de Procesos

- **Según su código.**

  - Reutilizables. Tienen datos asociados de manera que si vuelven a ejecutarse deben comenzar en su estado inicial.

  - Reentrantes. No tienen datos asociados, son "*puro código*".

- **Según su acceso al procesador.**

  - Apropiativos. Hacen uso del μP y no lo liberan hasta que terminan.

  - No apropiativos. Ceden en algún momento el uso del μP antes de terminar efectivamente.

- **Según su forma de ejecución.**

  - Residentes. Permanecen en la memoria durante toda su ejecución.

  - Intercambiables. Podrán ser llevados al área de *intercambio* en el disco.

2.8. Planificación de Procesos

Sus objetivos son: justicia, máxima capacidad de ejecución, máximo número de usuarios, predecibilidad, minimización de la carga, equilibrio en el uso de recursos, y seguridad en las prioridades.

2.8.1. Planificación a Corto Plazo

Los procesos pueden ser:

- **De políticas apropiativas.** Temporalmente suspenden la ejecución de un proceso para ejecutar otro.
- **De políticas no apropiativas.** Dejan que un proceso haga uso del μP ahsta que dicho proceso termine.