**SISTEMAS OPERATIVOS**

Es un conjunto de directrices y programas que gestionan los recursos de un computador, siendo un nexo de unión entre los componentes *hardware* y las aplicaciones *software* de más alto nivel.

* El programa que maneja las conexiones en sí es el *Kernel* (la pieza más importante al día de hoy del S.O)*.*

La presentación de un S.O se puede dar de forma gráfica (GUI) o mediante una consola de comandos (CLI).

**Clasificación – Estructura**

* Monolítico: un solo programa dividido en rutinas, de rápida ejecución para un uso específico, sin generalización. Si falla una rutina, el SO entero fallará (UNIX por ejemplo).
* Jerárquico (o multikernel): trabajan por niveles, cada uno con acceso a diferentes recursos. El sistema se fragmenta en operaciones más pequeñas, bien definidas y unidas entre sí. (Windows 10 por ejemplo).

**Clasificación – Servicios**

* Por número de usuarios: monousuario (MS-DOS) o multiusuario (Ubuntu)
* Por número de tareas simultáneas: monotarea (MS-DOS) o multitarea (Android)
* Por número de procesadores que puede gestionar: simétrico, los procesadores son tratados en equidad y hacen las mismas tareas del sistema. Asimétrico, hay un procesador maestro que realiza las tareas del sistema.

**Clasificación – Disponibilidad**

* Propietarios: requieren la adquisición de una licencia. Esto no da acceso al código y modificar el S.O es responsabilidad del usuario.
* Libres: permiten ser distribuidos, acceder al código fuente, modificarlos, etc. No confundir libre con gratuito

Clasificación – Licencia

* O.E.M (Original Equipment Manufacturer): las otorga el propietario del software al fabricante, optimizadas para ese hardware. La licencia pertenece al equipo no al usuario.
* Retail: comprada directamente al distribuidor. Un uso por máquina.
* Licencias por volumen.
* Licencias de educación.

**Clasificación – Arquitectura**

* La arquitectura del sistema operativo indicará la cantidad de memoria que puede direccionar.

32 bits (el sistema podrá direccionar hasta 232 bits de memoria, por un total de 4GB).

64 bits (el sistema podrá direccionar hasta 264 bits de memoria, por aproximadamente un total de 16 PB).

* Se puede observar que la arquitectura de 64 bits es varias magnitudes superiores. Por ello, la arquitectura de 32 bits está comenzando a desaparecer.

**Funciones**

El S.O debe cumplir 3 tareas críticas para la gestión de recursos de un ordenador. Estas son:

* Gestión de procesos: debe ser capaz de asignar tiempos de uso en CPU a los diferentes programas en ejecución.
* Gestión de memoria: debe gobernar de uso de la memoria principal (generalmente RAM) para que no se pierdan los datos de los programas en ejecución.
* Gestión de ficheros: debe ofrecer una visión opaca del almacenamiento no volátil, sin necesidad de conocer la forma lógica de organización usada de cara al usuario.

**Gestión de procesos**

* Ejecutar programa a programa de manera secuencial es ineficiente: imaginemos dos programas (A y B) que tardan 20 unidades de tiempo cada uno en ejecutarse. Eso serían 40 unidades de tiempo totales.
* La clave está en aprovechar la concurrencia en Entrada/Salida y los estados idílicos (o *idle*) del procesador: los procesos de Entrada/Salida son varios factores más lentos que la CPU.

Para que se aproveche los tiempos muertos del CPU, se gestionan estos por medio del CPU (método llamado *Multiprogramación*).

Hoy en día estas técnicas se aplican aun teniendo múltiples procesadores.

Hay dos tipos de limitaciones:

Trabajos limitados por procesos, es decir, más exigencia de CPU que de E/S.

Trabajos limitados por E/S, más exigencia en las operaciones de E/S que uso de CPU.

**Planificación de procesos**

Un S.O debe ser capaz de gestionar qué tiempos asignar a cada proceso.

Las tareas pueden acaparar el uso de CPU, por lo que el S.O tiene dos herramientas para expulsarlo:

* Temporizador
* Cuando se solicitan E/S

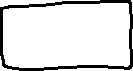
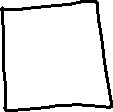
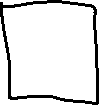
Se escoge el siguiente proceso que esté listo para ser ejecutado, sin esperar ni exigir el uso exclusivo del procesador.

**Problemas de concurrencia**

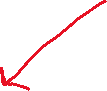
Al competir los procesos por el uso de recursos, los resultados pueden ser imprevisibles sin ninguna política de uso.

El S.O tiene que establecer mecánicas para la sincronización del uso de recursos (semáforos, pipes…)

* Condiciones de carrera (rare condition): cuando dos procesos intentan acceder al mismo recurso.



* Postergación: cuando los procesos no reciben tiempo suficiente de ejecución, por lo que nunca pueden terminar correctamente.
* Condición de Espera circular: cuando dos procesos piden recursos que está usando el otro, formando una cadena de espera indefinida.



***Flechas rojas: en espera que el banco de datos quede libre.***

***Flechas negras: en ejecución.***

**EJERCICIOS:**

**A:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t (tiempo)** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| **E/S** | **X** |  |  |  | **X** | **X** | **X** |  |  |  |  | **X** | **X** |  |  |  | **X** | **X** | **X** | **X** |
| **CPU** |  | **X** | **X** | **X** |  |  |  | **X** | **X** | **X** | **X** |  |  | **X** | **X** | **X** |  |  |  |  |

**B:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t (tiempo)** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| **E/S** | **X** | **X** |  |  |  | **X** |  | **X** | **X** | **X** |  |  |  | **X** | **X** | **X** |  |  | **X** | **X** |
| **CPU** |  |  | **X** | **X** | **X** |  | **X** |  |  |  | **X** | **X** | **X** |  |  |  | **X** | **X** |  |  |

**Pila simple**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t (tiempo)** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** |
| **E/S** | **A1 B1** | **B2** |  |  |  | **A5** | **A6** | **A7 B6** |  |  | **B8** | **B9** | **B10** | **A12** | **A13** |  |  | **B14** | **B15** | **B16 A17** | **A18** | **A19** | **A20 B19** | **B20** |
| **CPU** |  | **A2** | **A3** | **B3** | **A4** | **B4** | **B5** |  | **A8** | **B7** | **A9** | **A10** | **A11** | **B11** | **B12** | **A14** | **B13** | **A15** | **A16** |  | **B17** | **B18** |  |  |
| **Q=W=wait** |  |  | **B3** | **A4** | **B4** |  |  |  | **B7** | **A9** |  |  |  |  |  | **B13** | **A15** |  |  |  |  |  |  |  |

**C.P: A>B, S.C: PROCESOS PUEDEN EXPULSARSE**

#### GESTI***ÓN DE MEMORIA***

* En los sistemas monoprogramados, solo un único programa
* Sin embargo, en los sistemas multiprogramados, la memoria contiene la información de múltiples programas y sus datos
* Se observan dos problemas con en “lio” en la memoria: relocalización y la protección