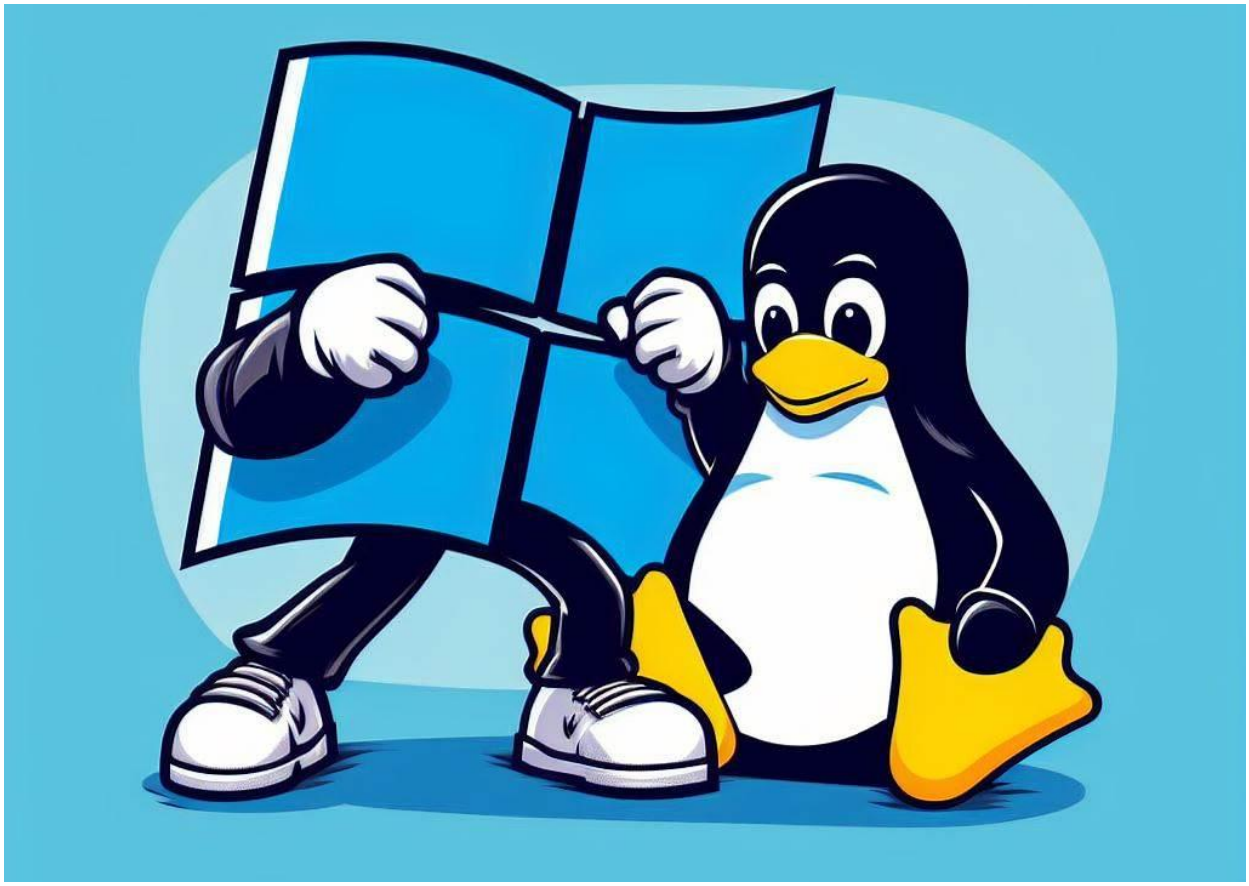
INTRODUCCIÓN A LOS SISTEMAS OPERATIVOS

INTRODUCCIÓN

Resumen teoría



Caporal Nicolás

Facultad de Informática

UNLP

2023

ÍNDICE

Clase 1

¿QUÉ ES UN SISTEMA OPERATIVO?

Un sistema operativo (SO) es **software**.

Como todo software tiene instrucciones y datos, por lo tanto necesita procesador (CPU) y memoria para ejecutarse.

En particular, el sistema operativo es un programa que controla la ejecución de los procesos y que actúa como interfaz (es decir, que hace de intermediario) entre el usuario de la computadora, y el hardware.

PERSPECTIVA DE ARRIBA HACIA ABAJO

Se ve al sistema operativo como una abstracción del hardware.

Es decir, una capa de software que oculta el hardware y la arquitectura al usuario, presentando a los programas de manera más simple.

Los programas que ejecutamos son “clientes” del SO. Lo que quiere decir que los programas le deben pedir servicio al sistema operativo para poder ejecutarse.

Se otorga comodidad y “amigabilidad” (friendliness)

PERSPECTIVA DE ABAJO HACIA ARRIBA

Se ve al sistema operativo como un administrador de recursos.

Entendiendo como “abajo” al hardware, el SO es el responsable de administrarlo. Maneja la CPU, la memoria, la tarjeta gráfica, así como también los dispositivos de entrada/salida (E/S) como el disco duro, una impresora, o unos auriculares.

También provee un conjunto de servicios a los usuarios del sistema (para poder comunicarse con el mundo exterior)

Para lograr una ejecución simultánea de los procesos, multiplexa (comparte) en el tiempo el uso de CPU y el espacio en memoria.

OBJETIVOS DE LOS SISTEMAS OPERATIVOS

COMODIDAD: Hacer más fácil y cómodo el uso del hardware (PC, servidor, router, controlador específico)

EFICIENCIA: Hacer uso más eficiente de los recursos del sistema.

CAPACIDAD DE EVOLUCIÓN: Permitir la introducción de nuevas funciones al sistema sin interferir con las funciones anteriores.
Un sistema operativo debe construirse y diseñarse teniendo en cuenta que el hardware y el software evoluciona, y el mismo sistema operativo también debe evolucionar para prestar mejoras en los servicios o nuevos servicios.

COMPONENTES DE UN SISTEMA OPERATIVO

El sistema operativo se puede ver como un conjunto de componentes que hacen a la solución final “Sistema Operativo”.

El componente principal es el **Kernel** (o núcleo). Este es el componente que está inmediatamente arriba del hardware e implementa todo lo necesario para poder utilizarlo.

Otro componente es la **Shell**. Esta es la capa más próxima al usuario. La shell es la manera de interactuar con el sistema operativo. En Windows, es el sistema de ventanas, en Linux la terminal de comandos.

También el sistema operativo viene con un conjunto de **herramientas**. Son programas extras con los que viene el sistema operativo que hacen a la completitud de este. Por ejemplo, cuando instalás Linux viene con el editor de texto Vi, y Windows trae una calculadora (entre muchas otras cosas).

EL KERNEL

El kernel es el componente principal del sistema operativo.

Es una porción de código del SO (recordemos que el SO es software).

Como el kernel es el componente que más se utiliza del SO, está siempre en memoria principal ya que está constantemente ejecutando.

Es quien se encarga de la administración de los recursos de hardware.
Implementa servicios esenciales como lo son el manejo de memoria, de CPU, la administración de procesos, la gestión de la E/S y la comunicación y la concurrencia.

SERVICIOS DE UN SISTEMA OPERATIVO

Administrar y planificar la CPU:

- Multiplexa (comparte o divide) la carga de trabajo
- Se busca imparcialidad en las decisiones
- Se intenta que no hayan bloqueos (que un proceso se detenga)
- Maneja las prioridades de los procesos

Administrar la memoria principal:

- Administrar el uso de memoria eficientemente
- Implementa un modelo de memoria virtual que se debe reflejar en lo físico
- Lleva una jerarquía de memoria
- Debe proteger la memoria para que no se pisen entre programas que compiten o que se ejecutan concurrentemente

Administrar el almacenamiento (File System):

- Administra el almacenamiento en memoria secundaria

Administrar dispositivos:

- Ocultar las dependencias del hardware
- Administrar los accesos simultáneos

Detectar errores y dar una respuesta:

- Detectar errores de hardware internos y externos
    - Errores de memoria/CPU
    - Errores de dispositivos
- Detectar errores de Software
    - Errores aritméticos (ej: división por 0)
    - Acceso no permitido a direcciones de memoria
- Evitar incapacidad del SO para conceder una solicitud de una aplicación

Encargarse de la interacción con el usuario:

- Implementar la Shell

Hacer contabilidad:

- Recoger estadísticas del uso del SO (ej: saber cuanta memoria ocupa un proceso)
- Monitorear parámetros de rendimiento
- Anticipar necesidades de mejoras futuras

COMPLEJIDAD

Un SO es un software extenso y complejo.

Se desarrolla por partes.

Cada una de estas partes deben ser analizadas y desarrolladas entendiendo su función, cuáles son sus entradas, y sus salidas.

Clase 2

FUNCIONES PRINCIPALES DE UN SISTEMA OPERATIVO

- Brindar abstracciones de alto nivel a los procesos del usuario
- Administrar eficientemente el uso de la CPU
- Administrar eficientemente el uso de la memoria
- Brindar asistencia para la realización de E/S por parte de los procesos

PROBLEMAS QUE UN SISTEMA OPERATIVO DEBE EVITAR

El sistema operativo en su rol administrador del hardware tiene la responsabilidad de proteger los recursos de posibles usos negativos, ya sean intencionales por parte de un programa malicioso o no intencionales generados por una mala programación.

Debe evitar:

- Que un proceso se apropie de la CPU
- Que un proceso intente ejecutar instrucciones de E/s
- Que un proceso intente acceder a una posición de memoria fuera de su espacio declarado

Para ello el sistema operativo, entre otras cosas debe:

- Gestionar el uso de la CPU
- Detener intentos de ejecución de instrucciones de E/S ilegales
- Detectar accesos ilegales a memoria
- Proteger el vector de interrupciones, así como las Rutinas de atención de interrupciones (RAI)

APOYO DEL HARDWARE

El sistema operativo, para poder lograr su objetivo y evitar los problemas anteriormente mencionados, necesita el apoyo del hardware.

La base para lograr esto, son las **interrupciones**.

A modo de repaso: las interrupciones interrumpen el secuenciamiento del procesador durante la ejecución de un proceso. Cuando ocurre una interrupción se apila la dirección de retorno, se accede al vector de interrupciones, allí se busca en un lugar determinado (según el ID de la interrupción), y se toma la dirección de memoria donde está la rutina para manejar esa interrupción específica.

Tres grandes características que el hardware debe proveer:

- **Modos de Ejecución:**

  Define limitaciones en el conjunto de instrucciones que se puede ejecutar en cada modo (modo kernel y modo usuario)

- **Interrupción de Clock:**

  Se debe evitar que un proceso se apropie de la CPU.

- **Protección de la Memoria:**

  Se deben definir límites de memoria a los que puede acceder cada proceso (registros base y registros límite)

MODOS DE EJECUCIÓN

La CPU lleva un bit que indica el modo en el que se está ejecutando. En función del modo actual la CPU puede hacer o no hacer ciertas cosas.

Las instrucciones privilegiadas deben ejecutarse en modo Kernel (o Supervisor)

En el modo kernel, se puede ejecutar cualquier instrucción, con los peligros que ello conlleva.

En modo Usuario, el proceso puede acceder sólo a su espacio de direcciones, es decir a las direcciones “propias”.

El kernel del SO se ejecuta en modo Kernel, el resto del SO (como la Shell) y los programas del usuario (como Chrome o Excel) se ejecutan en modo Usuario, que tiene un subconjunto de instrucciones permitidas.

Tener en cuenta que:

Cuando arranque el sistema, arranca con el bit en modo supervisor.

Cada vez que comienza a ejecutarse un proceso del usuario, este bit se DEBE PONER en modo usuario mediante una instrucción especial.

La única manera de pasar del modo Usuario al modo Kernel es que ocurra una interrupción. NO es el proceso de Usuario quien hace el cambio explícitamente. Cuando hay un trap (excepción o interrupción por software) o una interrupción, el bit de modo se pone en modo Kernel. Por lo tanto es prioritario para el SO operativo ser dueño y poder proteger el vector de interrupciones correctamente.

CÓMO ACTÚA EL HARDWARE EN LA PROTECCIÓN

Cuando el proceso de usuario intenta por sí mismo ejecutar instrucciones que pueden causar problemas (las llamadas instrucciones privilegiadas), el hardware lo detecta como una operación ilegal y le avisa al SO produciendo un trap.

RESUMIENDO MODOS DE EJECUCIÓN...

MODO KERNEL:

- **Gestión de procesos:** Creación y terminación, planificación, intercambio, sincronización y soporte para la comunicación entre procesos

- **Gestión de memoria:** Reserva de espacio de direcciones para los procesos, Swapping, Gestión y páginas de segmentos

- **Gestión E/S:** Gestión de buffers, reserva de canales de E/S y de dispositivos de los procesos

- **Funciones de soporte:** Gestión de interrupciones, auditoría, monitoreo

MODO USUARIO:

- Debug de procesos, definición de protocolos de comunicación, gestión de aplicaciones (compilador, editor, aplicaciones de usuario…)

- En este modo se llevan a cabo todas las tareas que no requieran accesos privilegiados

- En este modo no se puede interactuar con el hardware. El proceso trabaja en su propio espacio de direcciones

PROTECCIÓN DE LA CPU: INTERRUPCIÓN DE CLOCK

Para proteger la CPU se usa la interrupción por clock.

Constantemente y cada un tiempo determinado ocurre una interrupción, que “despierta” al sistema operativo para que controle y administre el sistema.

Que se “despierta” al sistema operativo es una manera sencilla de decir que se produce la interrupción, se cambia el bit de modo de la CPU pasando al modo Kernel, se accede al vector de interrupciones en la posición correspondiente a la interrupción por clock, se ejecuta la rutina de manejo la interrupción por clock, y recién ahí toma el control el Kernel, que hará lo que corresponda.

Se implementa a través de un clock y un contador en la CPU.

El Kernel le da valor al contador que se decrementa con cada tick de reloj y al llegar a cero, puede expulsar al proceso para ejecutar otro.

Así se evita que un proceso se apropie de la CPU.

PROTECCIÓN DE LA MEMORIA

El sistema operativo establece un límite en espacio de direcciones que puede usar un proceso. El proceso no puede salir de esos límites.

El SO es quien pone los procesos en memoria, por tanto sabe que por ejemplo Netbeans puede usar desde la dirección 15000h a la 20000h.

El kernel debe proteger para que los procesos del usuario no puedan acceder a donde no les corresponde (una zona de memoria protegida o el espacio de direcciones de otro proceso).

Sin embargo, mientras Netbeans se ejecuta, el SO no puede estar controlando que no se salga de los límites (necesitaría CPU), para esto la CPU tiene dos registros, llamados Registro Base y Registro Límite. Cuando el Kernel carga un proceso, también carga estos registros por medio de instrucciones privilegiadas (esta acción solo la puede realizar el kernel).

Entonces cada vez que un proceso intenta ejecutar una instrucción que accede a memoria, la CPU controla que esa dirección esté entre los límites correspondientes.
Si está entre los límites la ejecuta, si no está entre los límites, la CPU lanza una trap al SO.

PROTECCIÓN DE LA E/S

Las instrucciones de E/S se definen como privilegiadas y deben ejecutarse en modo Kernel.

Si un proceso de usuario desea realizar E/S, debe hacerlo a través de una llamada al sistema operativo (es un servicio del SO).

SYSTEM CALLS

El sistema operativo debe brindar una serie de servicios a los programas de usuario. Por ejemplo escribir o leer un archivo, ejecutar un programa o realizar E/S

Las System Calls son la forma en la que los programas de usuario acceden a los servicios del SO.

Son rutinas que los procesos pueden invocar para hacer uso de los servicios que el SO presta. Como toda función tiene sus parámetros (que pueden pasarse de varias maneras, por registros, por la pila, por memoria, etc).

Se ejecutan en modo Kernel.

Categorías de system calls:

- Control de Procesos
- Manejo de archivos
- Manejo de dispositivos
- Mantenimiento de información del sistema
- Comunicaciones

Cada SO implementa sus propias System Calls, por ejemplo el “mkdir” de Unix, se llama “CreateDirectory” en Windows. Este es el motivo por el que un programa hecho para un SO no funciona en otros.