Julia López - clase 7 - mochila del viajero

Sistema operativo
El sistema operativo es el soporte lógico que controla el funcionamiento del equipo físico. Es un software de comunicación usuario-dispositivo que comprende un conjunto de programas que ocultan los detalles del hardware y le ofrece al usuario una vía sencilla flexible para acceder al mismo. El mismo administra los recursos administrados por el hardware y actúa como intermediario entre computadora y usuario, ofreciéndole un ambiente más amigable y sencillo de interpretar. Empieza a funcionar en el momento en que encendemos nuestro dispositivo y deja de funcionar cuando lo apagamos.
Los OS están en gran parte de los dispositivos o aparatos tecnológicos que usan un microprocesador para funcionar.
Cada cosa que hacemos, como hacer un movimiento del mouse o un click, es informado por el SO al microprocesador. El sistema operativo administra los dispositivos de entrada y salida, la cola de procesos y los recursos del dispositivo.
Los servidores usan sistemas operativos en su mayoría derivados de UNIX, como Red Hat, o sistemas como Windows Server.
Los mismos tienen actualizaciones en las que incorporan mejoras para los usuarios.

Recursos administrados por el sistema operativo
• Gestionar la memoria de acceso aleatorio y ejecutar las aplicaciones, designando los recursos necesarios.
• Administrar la CPU, gracias al algoritmo de programación.
• Direccionar las entradas y salidas de datos (a través de drives), por medio de los periféricos de entrada y salida.
• Administrar la información para el buen funcionamiento de la PC.
• Dirigir las autorizaciones de uso para el usuario.
• Administrar los archivos.

Clasificación de los sistemas operativos

- Según sus usuarios:
  - **Monousuario**: normalmente computadoras domésticas. Solo un usuario puede ejecutar sus programas a la vez. Ej Windows doméstico hasta Me, DOS.
  - **Multiusuario**: varios usuarios están conectados al mismo tiempo, trabajando sobre el mismo núcleo. Permite que varios usuarios ejecuten simultáneamente sus programas ya sea por medio de terminales conectadas a la computadora o por sesiones remotas en una red de comunicación. Ej: Unix, Linux, Solaris. Windows a partir de XP
- Según su licencia:
  - **Open source**: permiten modificar, usar y adaptar a voluntad del usuario. Por ejemplo Ubuntu (distribución de Linux para escritorio y servidores) y Red Hat.
  - **Proprietary software**: no permiten modificaciones en el código.
- Según la gestión de tareas:
  - **Multitareas:** puede ejecutar varios procesos al mismo tiempo. Windows, Unix, Linux, Mac OSX.
  - **Monotarea:** sólo permite ejecutar un proceso por vez. Windows Me y Windows Vista?? DOS.
- Según la gestión de recursos:
  - **Centralizado**: solo permite utilizar los recursos de un solo ordenador. Windows, Unix, Linux, Mac OSX.

- **Distribuido**: permite ejecutar los procesos de más de un ordenador al mismo tiempo. Novell Netware, Windows Server, Cisco IOS, Unix, Linux.

- Según su estructura interna:
  - **Monolítica:** constituído por un solo programa, compuesto de una serie de rutinas entrelazadas. Suelen ser hechos a medida por lo que son muy rápidos pero no tienen flexibilidad. Ej VMS, Linux, Multics, Windows (hasta Me),
  - **Jerárquica:** cada parte del sistema tiene subpartes y está organizado en forma de niveles. Subdivide en capas o anillos perfectamente definidos y con clara interfaz con respecto al resto de los recursos. Unix, Multics
  - **Máquina virtual:** separan dos conceptos que están separados en otros sistemas, la multiprogramación y la máquina extendida. El objetivo es integrar distintos sistemas operativos dando la sensación de ser varias máquinas diferentes. Ej: Android. Microsoft Hyper-V, VMware, VirtualBox, QEMU, Kernel-Based Virtual machine.
  -**Cliente-servidor:** sirve para toda clase de aplicaciones. Es de propósito general. La idea es mantener la vision que tiene un usuario de un computador personal pero la red le permite compartir el espacio en disco o la impresora con el fin de economizar los recursos. No resuelven problemas de compartir información.

Generaciones de sistemas operativos

1. **Generación cero (década 1940):** las computadoras electrónicas digitales no tenían sistema operativo. Los programas, por lo regular, manejaban un bit a la vez, en columnas de switches mecánicos. Los programas de lenguaje máquina manejaban tarjetas perforadas.
2. **Primera Generación (1945-1955):** tubos de vacío y tableros enchufables. Se lograron construir máquinas calculadoras usando tubos de vacío. Estas máquinas eran enormes y ocupaban cuartos enteros con decenas de miles de tubos de vacío, pero eran mucho más lentas que incluso las computadoras personales más baratas de la actualidad. Toda la programación se realizaba en lenguaje de máquina absoluto.
3. **Segunda Generación (1955-1965):** Transistores y sistemas de lote. Estas máquinas se encerraban en cuartos de computadora con acondicionamiento de aire especial. Para ejecutar un programa, un programador escribía primero el programa en papel (en FORTRAN o ensamblador) y luego lo perforaba en tarjetas. Después, llevaba el grupo de tarjetas al cuarto de entrada y lo entregaba a uno de los operadores. Cuando la computadora terminaba el trabajo que estaba ejecutando en ese momento, se separaba la salida impresa y se llevaba al cuarto de salida donde el programador podía buscarla. Luego, el operador tomaba uno de los grupos de tarjeta traídos del cuarto de entrada y lo introducía en el lector. Si se requería el compilador de FORTRAN, el operador tenía que traerlo de un archivero e introducirlo en el lector. Dado el alto costo del equipo, la solución que se adoptó generalmente fue el sistema por lotes. El principio de este modo de operación consistía en juntar una serie de trabajos en el cuarto de entrada, leerlos y grabarlos en una cinta magnética usando una computadora pequeña y (relativamente) económica. Después de cerca de una hora de reunir un lote de trabajos, la cinta se rebobinaba y se llevaba al cuarto de la máquina, donde se montaba en una unidad de cinta. El operador cargaba entonces un programa especial, que leía el primer trabajo de la cinta y lo ejecutaba. La salida se escribía en una segunda cinta, en lugar de imprimirse. Cada vez que terminaba un trabajo, el sistema operativo leía automáticamente el siguiente trabajo de la cinta y comenzaba a ejecutarlo.

4. **Tercera generación (1965-1970):** Circuitos integrados (CI) y multiprogramación. Las máquinas diferían sólo en el precio y el rendimiento (memoria máxima, velocidad del procesador, número de dispositivos de E/S permitidos, entre otros). IBM introdujo System/360. Todas las máquinas tenían la misma arquitectura y conjunto de instrucciones asi que los programas escritos para una máquina podían ejecutarse en todas. Los 360 y los sistemas operativos de tercera generación popularizaron la multiprogramación. El problema era el tiempo de espera, por lo que se dividió la memoria en varias secciones, con un trabajo distinto en cada partición. Mientras un trabajo estaba esperando que terminara su E/S, otro podía estar usando la CPU. Si se podían tener en la memoria principal suficientes trabajos a la vez, la CPU podía mantenerse ocupada casi todo el tiempo. También, tenían la capacidad de leer trabajos de las tarjetas al disco tan pronto como se llevaban al cuarto de computadoras. Luego, cada vez que un trabajo terminaba su ejecución, el sistema operativo podía cargar uno nuevo del disco en la partición que había quedado vacía y ejecutarlo.
5. **Cuarta generación (1980 - actualidad):** Con la invención de los circuitos integrados a gran escala (LSI), chips que contienen miles de transistores en un cm2 de silicio, nació la era de la computadora personal. Dos sistemas operativos dominaron inicialmente el campo de las computadoras personales y las estaciones de trabajo: MS-DOS de Microsoft (en la IBM PC y otras basadas en la CPU Intel 8088) y UNIX. Más tarde, la Pentium y Pentium Pro. El sucesor de Microsoft para MS-DOS fue Windows que se ejecutaba encima, pero a partir de 1995 se produjo una versión autosuficiente de WINDOWS. UNIX, que domina en las estaciones de trabajo y otras computadoras del extremo alto, como los servidores de red. UNIX es popular sobre todo en máquinas basadas en chips RISC de alto rendimiento.

Llamadas al sistema
Son la manera en la cual un programa solicita una acción al sistema operativo con el que interactúa. Esta acción es el punto de enlace entre el modo usuario y el modo privilegiado del SO. Son lo que permite a las app usar el hardware. Su objetivo es diferenciar las acciones que el usuario puede hacer y las que no. Solo le deja hacerlo en modo privilegiado. El SO autoriza o administra todas las acciones potencialmente riesgosas.

Clasificación de llamadas al sistema
Si bien todas trabajan como unidades de control para el sistema operativo, se establecieron cinco tipos:

1. **Gestión de control:** supervisa el inicio, creación, detención y finalización de los procesos.
2. **Gestión de archivos:** incluyen la creación, eliminación, apertura, cierre, escritura y lectura de archivos.
3. **Gestión de dispositivos:** administra los recursos disponibles, como el almacenamiento.
4. **Gestión de información:** asegura la puntualidad e integridad de la información.
5. **Comunicación entre procesos:** coordina la interacción entre los distintos procesos y aplicaciones.

El **kernel** es el cerebro del sistema operativo y se encuentra alojado en él. Es el encargado de actuar entre las diferentes aplicaciones y sus necesidades con los recursos que posee el dispositivo para ejecutarlos. Ej: al hacer click para guardar un documento el kernel interactúa con la memoria secundaria para resguardar la información.

El kernel decide cuándo asignar o quitar recursos de hardware a las apps que se ejecutan en el software y asigna prioridades según las necesidades del SO.
Esas interacciones se llevan a cabo a través de llamadas al sistema. Las llamadas son el método que tienen las app para solicitar un servicio o un recurso. Ej: solicitar a la impresora una impresión.
Windows, Linux y Mac poseen su propio kernel pero hay otros sistemas que se basan en kernels ya creados como Android, que se basa en el kernel de Linux.

Clasificación de kernels

- monolítico: Linux trabaja con este. Es un código de muchas líneas que está alojado en un solo espacio de memoria y posee todo lo que necesita, es decir, los drivers, los servicios y los métodos de administración de recursos. La desventaja es que se desperdicia mucho espacio en memoria porque al cargar el kernel se carga todo (el 70% no se usa). Es más veloz porque se comunica con llamadas al sistema. Si un sistema falla, todo el núcleo falla.
- microkernel: solo posee las instrucciones básicas de administración en un pequeño espacio de memoria y deja a los diferentes dispositivos su propio manejo. Se encarga solo de las tareas más básicas. La desventaja es que un microkernel pertenece únicamente a un dispositivo entonces hay que diseñar un SO para cada dispositivo. Es más lento que el monolítico porque se comunica por paso de mensajes y requiere muchas líneas de código. Pero es más fácil agregar nuevas funcionalidades.
- Kernel híbrido: es como un microkernel pero agrega un poco más código no esencial. Lo hace más rápido y es compatible con un gran número de dispositivos.
- Nanokernel: el código es más reducido que en el microkernel pero más dificil de crear. Se comunica con paso de mensajes, permite una fácil modificación del sistema operativo.

Cuando un dispositivo falla, la función del kernel es detener todo lo que está haciendo la computadora para evitar un daño en el SO. Todas las acciones de la computadora pasan por el kernel.