Sistemas Operativos

Sistemas de cómputo

Hemos visto la evolución de los sistemas de cómputo desde el punto de vista del hardware, y cómo llegaron a soportar varios usuarios corriendo varias aplicaciones, todo sobre un mismo equipamiento.

Sistemas de cómputo

Ahora veremos de qué manera evolucionó el software asociado a esos sistemas de cómputo para permitir que esos diferentes usuarios y esas diferentes aplicaciones pudieran compartir el hardware sin ocasionarse problemas unos a otros, y obteniendo el máximo rendimiento posible del equipamiento.

La pieza que falta en este complejo mecanismo es el sistema operativo, un software básico cuya función principal es la de ser intermediario entre los usuarios y el hardware del sistema de cómputo.

Evolución del software de base

Las primeras computadoras estaban dedicadas a una única tarea, perteneciente a un único usuario. Podían ser utilizadas por diferentes usuarios, pero cada uno debía esperar su turno para reprogramarlas manualmente, lo cual era laborioso y se llevaba gran parte del tiempo por el cual esos usuarios pagaban.

Una vez que se popularizaron las máquinas de programa almacenado, se pudo minimizar el tiempo ocioso adoptando esquemas de carga automática de trabajos. Un trabajo típico consistía en la compilación y ejecución de un programa, o la carga de un programa compilado más un lote de datos de entrada, y la impresión de un cierto resultado de salida del programa. Estos trabajos estaban definidos por lotes de tarjetas perforadas, de ahí su nombre de trabajos **por lotes** o, en inglés, **batch**.

Más adelante, conforme las tecnologías permitían ir aumentando la velocidad de procesamiento, se notó que los procesadores quedaban desaprovechados gran parte del tiempo debido a la inevitable actividad de entrada/salida. Así se idearon sistemas que optimizaban la utilización de la CPU, al poderse cargar más de un programa en la memoria y poder conmutar el uso del procesador entre ellos. Éstos fueron los primeros **sistemas multiprogramados**.

Una vez que llegó la posibilidad de tener varios programas coexistiendo simultáneamente en la memoria, se logró que la conmutación del uso del procesador fuera tan rápida, que pareciera que cada programa funcionaba sin interrupciones.

Aunque el sistema era de **tiempo compartido**, el usuario utilizaba la computadora como si estuviera dedicada exclusivamente a correr su programa. Así los sistemas multiprogramados se volvieron interactivos.

Todas éstas fueron innovaciones de software, y fueron estableciendo principios y técnicas que serían adoptadas en lo sucesivo. Con la llegada de la computación personal, los sistemas de cómputo eran de capacidades modestas. Los **sistemas operativos** que permitían la ejecución de aplicaciones de los usuarios en estos sistemas de cómputo comenzaron pudiendo correr una sola aplicación por vez y de un solo usuario; es decir, se trataba de sistemas **monotarea** y **monousuario**.

Sin embargo, con la industria de las computadoras personales y la del software para computadoras personales traccionándose una a la otra, aparecieron sistemas operativos **multiusuario** y **multitarea**, sumamente complejos, que se convirtieron en un nuevo terreno para ensayar y mejorar las tecnologías de software y hardware.

Componentes del SO

Los modernos sistemas operativos tienen varios componentes bien diferenciados. El componente que constituye el sistema operativo propiamente dicho es el llamado **núcleo** o **kernel**. Junto al kernel es habitual encontrar un conjunto de **programas utilitarios o software de sistema**, que no es parte del sistema operativo, estrictamente hablando, pero que en general es indispensable para la administración y mantenimiento del sistema. También se encuentra junto a este software del sistema alguna forma de **interfaz de usuario**, que puede ser gráfica o de caracteres. Esta interfaz de usuario se llama en general **shell**, especialmente cuando la interfaz es un procesador de comandos, basado en caracteres, y los comandos se tipean.

Hay algunas excepciones a esta estructura de componentes, por ejemplo, en los sistemas operativos **empotrados**, que están ligados a un dispositivo especial y muy específico, como es el caso de algunos robots, instrumental médico, routers, electrodomésticos avanzados, etc. Estos sistemas operativos constan de un kernel que tiene la misión de hacer funcionar cierto hardware, pero no necesariamente incluyen una interfaz de usuario (porque el usuario no necesita en realidad comunicarse con ellos) o no incluyen software de sistema porque sus usuarios no son quienes se encargan de su mantenimiento. Pero los sistemas operativos de propósito general integran una **distribución** que normalmente contiene a estos tres componentes.

Software involucrado

Un típico sistema operativo multipropósito, actual, debe dar soporte entonces a la actividad de una gan variedad de aplicaciones. No solamente a la interfaz de usuario o procesador de comandos, más el software de sistema incluido,

sino también a toda la gama de aplicaciones que desee ejecutar el usuario, como programas de comunicaciones (navegadores, programas de transferencia de archivos, de mensajería); aplicaciones de desarrollo de programas (compiladores, intérpretes de diferentes lenguajes).

Kernel o núcleo

El **kernel** o núcleo es esencialmente un conjunto de rutinas que permanecen siempre residentes en memoria mientras la computadora está operando. Estas rutinas intervienen en todas las acciones que tengan que ver con la operación del hardware.

El kernel funciona no solamente como un mecanismo de administración y control del hardware o conjunto de recursos físicos, sino también de ciertos recursos del sistema que son lógicos, como los archivos. Además tiene la capacidad de poner en ejecución a los programas que se encuentran en el sistema. Cuando un programa está en ejecución, lo llamamos un **proceso**. El sistema operativo controla la creación, ejecución y finalización de los procesos.

El kernel ofrece su capacidad de control de todos los recursos a los procesos o programas en ejecución, quienes le solicitan determinadas operaciones sobre esos recursos. Por ejemplo, un proceso que necesita utilizar un dispositivo de entrada/salida, o un recurso lógico como un archivo, hace una petición o llamada al sistema solicitando un servicio al sistema operativo. El servicio puede tratarse de una operación de lectura, escritura, creación, borrado, etc. El sistema operativo centraliza y coordina estas peticiones de forma que los procesos no interfieran entre sí.

Si los procesos de usuario pudieran utilizar los recursos en cualquier momento y sin coordinación, los resultados podrían ser desastrosos. Por ejemplo, si dos o más programas quisieran usar la impresora al mismo tiempo, en el papel impreso se vería una mezcla de las salidas de los programas que no serviría a ninguno de ellos.

Como el sistema operativo debe coordinar el acceso de los diferentes procesos a esos recursos, resulta necesario que cuente con alguna forma de imponer conductas y límites a esos usuarios y programas, para evitar que ellos tomen control del sistema en perjuicio de los demás. Para garantizarle este poder por sobre los usuarios, el sistema operativo requiere apoyo del hardware: su código se ejecuta en un modo especial, el **modo privilegiado** del procesador.

Modo de ejecución dual

Los modernos procesadores funcionan en lo que llamamos **modo dual** de ejecución, donde el ISA se divide en dos grupos de instrucciones. Ciertas instrucciones que controlan el modo de operación de la CPU, el acceso a memoria, o a las

unidades de Entrada/Salida, pertenecen al grupo de instrucciones del **modo privilegiado**. Un programa de usuario que se está ejecutando funciona en modo **no privilegiado**, donde tiene acceso a la mayoría de las instrucciones del ISA, pero no a las instrucciones del modo privilegiado.

El programa en ejecución ejecutará instrucciones en modo no privilegiado hasta que necesite un servicio del sistema operativo, como el acceso a un recurso físico o lógico.

Modo de ejecución dual

Para requerir este servicio, el programa ejecuta una instrucción de **llamada al sistema** o **system call**, que es la única instrucción del conjunto no privilegiado que permite a la CPU conmutar al modo privilegiado.

Modo de ejecución dual

La llamada al sistema conmuta el modo de la CPU a modo privilegiado **y además** fuerza el salto a una cierta dirección fija de memoria donde existe código del kernel. En esa dirección de memoria existe una rutina de atención de llamadas al sistema, que determina, por el contenido de los registros de la CPU, qué es lo que está solicitando el proceso.

Con estos datos, esa rutina de atención de llamadas al sistema dirigirá el pedido al subsistema del kernel correspondiente, ejecutando siempre en modo privilegiado, y por lo tanto, con completo acceso a los recursos.

Modo de ejecución dual

Modo de ejecución dual

El subsistema que corresponda hará las verificaciones necesarias para cumplir el servicio:

- El usuario dueño del proceso, ¿tiene los permisos necesarios?
- El recurso, ¿está disponible o está siendo usado por otro proceso?, etc.

Cuando se cumplan todos los requisitos, se ejecutará el servicio pedido y luego se volverá a modo usuario, a continuar con la ejecución del proceso.

Aplicaciones

Cualquier aplicación que funcione en el sistema, ya sean las de sistema o generadas por el usuario, se ubica en igualdad de condiciones con las demás.

Bibliotecas

Todas las aplicaciones, en algún momento, requieren funciones que ya están preparadas para su uso en **bibliotecas** especializadas en algún sentido. La vinculación de los programas de usuario con las bibliotecas puede hacerse al tiempo de compilación o, cuando las bibliotecas son **de carga dinámica**, al tiempo de ejecución.

Kernel

Al ejecutarse los procesos, las bibliotecas, para completar su funcionamiento, normalmente necesitan recurrir a servicios del kernel. Los diferentes subsistemas del kernel se ocupan de cada clase de servicios y de manejar diferentes clases de recursos.

Llamadas al sistema

La comunicación entre los procesos de usuario y sus bibliotecas, por un lado, y el kernel y sus subsistemas, por otro, se produce cuando ocurre una llamada al sistema o system call. Es en este momento cuando se cruza el límite entre modo usuario y modo privilegiado, o espacio de usuario y espacio del kernel.

Cronología

Entre la década de 1960 y principios del siglo XXI surgieron gran cantidad de innovaciones tecnológicas en el área de sistemas operativos. Muchas de ellas han tenido éxito más allá de los fines experimentales y han sido adoptadas por sistemas operativos con gran cantidad de usuarios. Diferentes sistemas operativos han influido en el diseño de otros posteriores, creándose así líneas genealógicas de sistemas operativos.

Es interesante seguir el rastro de lo que ocurrió con algunos sistemas importantes a lo largo del tiempo, y ver cómo han ido reconvirtiéndose unos sistemas en otros.

Cronología

Por ejemplo, el sistema de archivos diseñado para el sistema operativo CP/M de Digital, en los años 70, fue adaptado para el MS-DOS de Microsoft, cuya evolución final fue **Windows**. Los diseñadores de Windows NT fueron los mismos que construyeron el VMS de los equipos VAX, también de Digital, y aportaron su experiencia. De hecho, muchas características de la gestión de procesos y de entrada/salida de ambos sistemas son idénticas.

Cronología

Otra importante línea genealógica es la que relaciona el antiguo Multics, por un lado, con **Unix** y con Linux; y más recientemente, con el sistema para plataformas móviles Android. Unix fue el primer sistema operativo escrito casi totalmente en un lenguaje de alto nivel, el **C**, lo cual permitió portarlo a diferentes arquitecturas. Esto le dio un gran impulso y la comunidad científica lo adoptó como el modelo de referencia de los sistemas operativos de tiempo compartido.

En 1991 **Linus Torvalds** lanzó un proyecto de código abierto dedicado a la construcción de un sistema operativo compatible con Unix pero sin hacer uso de ningún código anteriormente escrito, lo que le permitió liberarlo bajo una licencia libre. La consecuencia es que Linux, su sistema operativo, rápidamente atrajo la atención de centenares de desarrolladores de todo el mundo, que sumaron sus esfuerzos para crear un sistema completo y libremente disponible.

Linux puede ser estudiado a fondo porque sus fuentes no son secretos, como en el caso de los sistemas operativos propietarios. Esto lo hace ideal, entre otras cosas, para la enseñanza de las Ciencias de la Computación. Esta cualidad de sistema abierto permitió que otras compañías lo emplearan en muchos otros proyectos.

Cronología

De la misma manera, otra empresa de productos de computación de notable trayectoria, **Apple**, produjo un sistema operativo para su línea de computadoras personales Macintosh. Su sistema MacOS estaba influenciado por desarrollos de interfaces de usuario gráficas realizadas por otra compañía, Xerox, y también derivó en la creación de un sistema operativo para dispositivos móviles.

Otros sistemas operativos han cumplido un ciclo con alguna clase de final, al no superar la etapa experimental, haberse transformado definitivamente en otros sistemas, desaparecer del mercado o quedar confinados a cierto nicho de aplicaciones. Algunos, por sus objetivos de diseño, son menos visibles, porque están destinados a un uso que no es masivo, como es el caso del sistema de tiempo real **QNX**.

Servicios del SO

Ejecución de programas

Ejecución de programas

Estados de los procesos

Estados de los procesos

Estados de los procesos

Estados de los procesos

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos concurrentes

Procesos paralelos

Procesos

Procesos

Comando top

Comandos de procesos

Gestión de archivos

Gestión de archivos

Árbol de directorios

Sistema de archivos

Sistema de archivos

Sistema de archivos

Sistema de archivos

El proceso P1 tiene asignadas cuatro páginas (de las cuales sólo la página 2 está presente en memoria principal), y P2, dos páginas (ambas presentes). Hay tres marcos libres (M4, M6 y M7) y la zona de intercambio está vacía.

Swapping o intercambio

P1 recibe la CPU y en algún momento ejecuta una instrucción que hace una referencia a una posición dentro de su página 3 (que no está en memoria).

Swapping o intercambio

Ocurre una falta de página que trae del almacenamiento la página 3 de P1 a un marco libre. La página 3 se marca como presente en la tabla de páginas de P1.

Swapping o intercambio

Enseguida ingresa P3 al sistema y comienza haciendo una referencia a su página 2.

Swapping o intercambio

Como antes, ocurre una falta de página, se trae la página 2 de P3 del disco, y se copia en un marco libre. Se marca la página 2 como presente y P3 continúa su ejecución haciendo una referencia a una dirección que queda dentro de su página 3.

Swapping o intercambio

Se resuelve como siempre la falta de página para la página 3 y P3 hace una nueva referencia a memoria, ahora a la página 4.

Swapping o intercambio

Pero ahora la memoria principal ya no tiene marcos libres. Es el momento de elegir una página víctima para desalojarla de la memoria. Si la página menos recientemente usada es la página 2 de P1, es una buena candidata. En caso de que se encuentre modificada desde que fue cargada en memoria, se la copia en la zona de intercambio para no perder esas modificaciones, y se declara libre el marco M2 que ocupaba.

Swapping o intercambio

Se marca como no presente la página que acaba de salir de la memoria principal.

Swapping o intercambio

Se copia la página que solicitó P3 en el nuevo marco libre, se la marca como presente en la tabla de páginas de P3, y el sistema continúa su operación normalmente.

Memoria y protección

Memoria y protección

Memoria y protección

Memoria y protección

Este emulador de PC construido en Javascript nos permite practicar los comandos del shell dentro de una *máquina virtual* Linux y desde el navegador, sin necesidad de una instalación completa en nuestro equipo.

- Para retroceder en el terminal
- Teclas Ctrl-Up, Ctrl-Down, Ctrl-PageUp y Ctrl-PageDown.
- Para copiar datos a la máquina virtual
- Copiar el texto a la caja o **clipboard** a la derecha.
- En el shell de la máquina virtual escribir: cat < /dev/clipboard > /tmp/archivo
- Ahora tenemos ese mismo texto en el archivo /tmp/archivo.
- Para extraer datos de la máquina virtual
- Invertir el procedimiento anterior: cat mi_archivo > /dev/clipboard
- Seleccionar el texto en la caja **clipboard** y copiarlo a alguna otra aplicación en nuestro equipo, tal como un editor.

Referencias