Apuntes de Sistemas Operativos

Daniel Araya Román

2023-05-30

1. Introducción

Un **sistema operativo** es un programa que administra el hardware de una computadora. Actúa como intermediario entre el usuario y el hardware. Un aspecto sorprendente de los sistemas operativos es la gran variedad de formas en que llevan a cabo estas tareas. Los sistemas operativos *mainframe* están diseñados para optimizar el uso del hardware. Algunos están diseñados para ser prácticos, otros para ser eficientes y otros para ser ambas cosas. Antes de adentrarnos en los detalles de los sistemas operativos, es importante entender acerca de la estructura del sistema. Dado que un sistema operativo es un software grande y complejo, debe crearse pieza por pieza. En este capítulo se describe los principales componentes de un sistema operativo.

1.1 Qué hace un sistema operativo?

Un sistema operativo es un sistema informático que puede dividirse en cuatro componentes: *hardware, sistema operativo, programas de aplicación y usuarios.* El **hardware**, la **unidad central de procesamiento** (CPU), la **memoria** y los **dispositivos de entrada/salida** (E/S), proporcionan los recursos básicos de cómputo al sistema. Los **programas de aplicación**, como los procesadores de texto, las hojas de cálculo, los compiladores y los navegadores web, definen las formas en que estos recursos se emplean para resolver los problemas informáticos de los usuarios.

Analogía

Sistema operativo → Gobierno → Entorno de programas → Trabajo útil

1.1.1 Punto de vista del usuario

La mayoría de usuarios disponen de un monitor, teclado, un ratón, una unidad de sistema. Un sistema así se diseña para que el usuario **monopolice** sus recursos. El objetivo es maximizar el trabajo que el usuario realice. En este caso tiene que diseñarse para que sea de fácil uso.

En otros casos, un usuario se sienta en frente a un terminal conectado a un **mainframe** o una **microcomputadora**. Otros usuarios acceden simultáneamente a través de otros terminales. Estos usuarios comparten recursos y pueden intercambiar información. En tal caso, el sistema operativo se diseña para maximizar la utilización de recursos, de modo que cada usuario disponga sólo de una parte equitativa que le corresponde.

En otros casos, los usuarios usan **estaciones de trabajo** conectadas a redes de otras estaciones de trabajo y servidores. Los usuarios tienen recursos dedicados, pero también tienen recursos compartidos como la red y los servidores. Por tanto su sistema operativo está diseñado para llegar a un compromiso entre la usabilidad individual y la utilización de recursos.

1.1.2 Vista del sistema

El sistema operativo es el programa más íntimamente relacionado con el hardware. Podemos ver al sistema operativo como un **asignador de recursos**. El sistema operativo **actúa** como el administrador de estos recursos. Al enfrentarse a numerosas y posibles conflictivas solicitudes de recursos, el sistema operativo debe de decidir cómo asignarlos a programas y usuarios específicos, de modo que el computador opere de manera eficiente y equitativa. Un punto de vista que difiere al sistema operativo, hace hincapié en la necesidad de controlar dispositivos de E/S y programas de usuario. Un sistema operativo es un **programa de control**.

1.1.3 Definición de sistemas operativos

No hay una definición de sistema operativo que sea completamente adecuada. Estos existen porque ofrecen una forma razonable de resolver el problema de crear un sistema informático utilizable. El **objetivo principal** de las computadoras es **ejecutar** programas de usuario y resolver problemas del mismo fácilmente. Con este objetivo se construye el hardware de la computadora. Ya que el hardware no es fácil de usar, se desarrollaron programas de aplicación. Estos programas requieren operacaiones comunes, y estas se incorporan en una pieza de software que es el sistema operativo. Además, no hay ninguna definición universalmente aceptada sobre qué forma parte de un sistema operativo. Las características varían de un sistema a otro. Algunos sistemas operativos ocupan 1 megabyte de espacio y no proporcionan ni un editor a pantalla completa, mientras que otros necesitan gigabytes de espacio y están completamente basados en sistemas gráficos de ventanas.

Unidades de medidad

$$
\begin{aligned}
1 \text{ bit} &= 0 \text{ or } 1 \\
1 \text{ byte} &= 8 \text{ bits} \\
1 \text{ kilobyte} &= 1024^1 \text{ bytes} \\
1 \text{ megabyte} &= 1024^2 \text{ bytes} \\
1 \text{ gigabyte} &= 1024^3 \text{ bytes}
\end{aligned}
$$

Otra definición común es que un sistema operativo es aquel programa que se ejecuta continuamente en la computadora (usualemente denominado ***kernel***), siendo todo lo demás programas del sistema y programas de aplicación.

1.2 Organización de una computadora

Antes de entender como funciona una computadora, debemos entender su **estructura**.

1.2.1 Funcionamiento de una computadora

Una computadora moderna de propósito general consta de una o más CPU y una serie de controladoras de dispositivo conetadas a través de un **bus común** que proporciona acceso a la **memoria compartida**. Cada controladora de dispositivo se encarga de un tipo específico de dispositivo, por ejemplo, unidades de disco, dispositivos de audio y pantallas de video. La CPU y estas controladoras pueden funcionar de forma concurrente, compitiendo por los ciclos de memoria. Para asegurar el acceso de forma ordenada a la memoria compartida, se proporciona una controladora de memoria cuya función es **sincronizar** el acceso a la misma.

Para que una computadora empiece a funcionar, es necesario que tenga un programa de inicio de ejecutar. Este **programa de arranque** suele ser simple. Normalmente se almacena en la memoria **ROM (read-only memory)** o en una memoria **EEPROM (electrically erasable programmable read-only memory)**; conocida con el término general de **firmware**. El programa de arranque debe saber cómo cargar el sistema operativo e iniciar la ejecución del mismo. Para esto debe localizar y cargar en memoria el kernel (núcleo) del sistema operativo. Después, el sistema operativo comienza ejecutando el **primer proceso**, como por ejemplo *init* y espera a que se produzca algún suceso.

La ocurrencia de un suceso normalmente se indica mediante una **interrupción**, bien del hardware o software. El hardware puede activar una interrupción en cualquier instante enviando una señal a la CPU, normalmente a través del bus del sistema. El software puede activar una interrupción mediante una operación especial llamada de **llamada al sistema**. Cuando la CPU se interrumpe, deja lo que está haciendo e inmediatamente transfiere la ejecución a una posición fijada. Normalmente contiene la dirección de inicio donde se encuentra la rutina de servicio a la interrupción. Luego de ejecutar esta rutina la CPU reanuda la operación que estuviera haciendo.

El método más simple para tratar la transferencia consiste en invocar una rutina genérica para examinar la información de la interrupción. Sin embargo estas interrupciones deben de tratarse rápidamente, y este procedimiento es algo lento. Solamente solo es posible un número predefinido de interrupciones; o usar otro sistema consistente en disponer una tabla de punteros a las rutinas de interrupción. Proporcionando la velocidad necesaria, de forma indirecta se llama a través de la tabla. Sin necesidad de una rutina intermedia, generalmente la tabla de punteros se almacena en una posición inferior de la memoria. Este **vector de interrupciones**, se indexa mediante un número uníco que se proporciona con la solicitud de interrupción, para obtener la dirección de la rutina de servicio a la interrupción para el dispositivo correspondiente. Sistemas operativos tan diferentes como **Windows y UNIX** manejan las interrupciones de esta forma.

Esta arquitectura debe de almacenar la dirección de la instrucción interrumpida. Las arquitectuas más recientes almacenan la dirección de retorno en la **pila del sistema**. Si la rutina e interrupción necesita modificar el estado del procesador, debe guardar explícitamente el estado actual y luego restaurar dicho estado antes de volver. Después la dirección de retorno guardada se carga en el contador de programa y la ejecución continúa como si nada hubiera pasado.

1.2.2 Estructura de almacenamiento

Los programas de una computadora deben hallarse en la memoria principal (memoria **RAM**), *random-access memory*, para ser ejecutados. El procesador puede acceder directamente. Habitualmente, se implementa con una tecnología de semiconductores denominada **DRAM (dynamic random-access memory)**, que forma una matriz de palabras de memoria. Cada palabra tiene su propia dirección, se interactúa a través de una secuencia de carga (load) o almacenamiento (store) de instrucciones en direcciones específicas de memoria. La instrucción load mueve una palabra desde la memoria principal a un registro interno de la CPU, mientras que la instrucción store mueve una palabra desde el registro interno a la memoria principal.

Un ciclo típico **instrucción-ejecución**, cuando se ejecuta en un sistema de la arquitectura de **Von Neumann**, primero se extrae una instrucción de memoria y se almacena dicha instrucción en el **registro de instrucciones**. Continuamente la instrucción se decodifica y puede dar lugar a que se extraigan los operandos de la memoria y se almacenen en algún registro interno. Después de ejecutar la instrucción con los operandos necesarios, el resultado se almacena de nuevo en memoria.

Naturalmente la unidad de memoria solo ve un flujo de direcciones de memoria; no sabe cómo se an generado o qué son. Se puede ignorar cómo genera un programa una dirección de memoria. Solo interesa la secuencia de direcciones de memoria generadas por el programa en ejecución. Idealmente, es deseable que los programas y los datos residan en la memoria principal de forma permanente. Normalmente no es posible por dos razones:

1. Normalmente, la memoria principa es demasiado pequeña para contener todos los programas y datos necesarios de forma permanente.

2. La memoria principal es volátil; pierde su contenido cuando se le quita la alimentación.

Por tanto la mayor parte de los sistemas informáticos requieren de almacenamiento secundario, como una extensión de la memoria principal. Este puede almacenar grandes cantidades de datos de manera permanente. El dispositivo de almacenamiento secundario más común es el **disco magnético**, proporcionando un sistema de almacenamiento tanto para programas como para datos. La mayoría de los programas se almacenan en un disco hasta que se cargan en memoria. Por lo que una apropiada administración del almacenamiento en disco es de importancia crucial en un sistema informático como se puede ver en el *capítulo 12*.

No obstante, en un sentido amplio, la estructura de almacenamiento consta de registros, memoria principal, y discos magnéticos, solo es uno de los muchos sistemas de almacenamiento. Otros sistemas pueden incluir memoria caché, CD-ROM, cintas magnéticas, etc. Cada sistema proporciona las funciones básicas para guardar datos y mantenerlos hasta que estos sean recuperados en un instante posterior. Las principales diferencias entre los sistemas de almacenamiento son la **velocidad**, el **coste**, el **tamaño** y la **volatilidad**.

1.2.3 Estructura de E/S