Arquitectura y Sistemas Operativos

Presentación de la Cátedra: Integrantes

Cronograma de la materia:

- Formas de Evaluación
- Materiales y Recursos (Campus CVG)

Formas de Evaluación

- **Asistencia -> a definir**
- **Aprobación de parciales teórico-práctico y trabajos prácticos -> a definir**

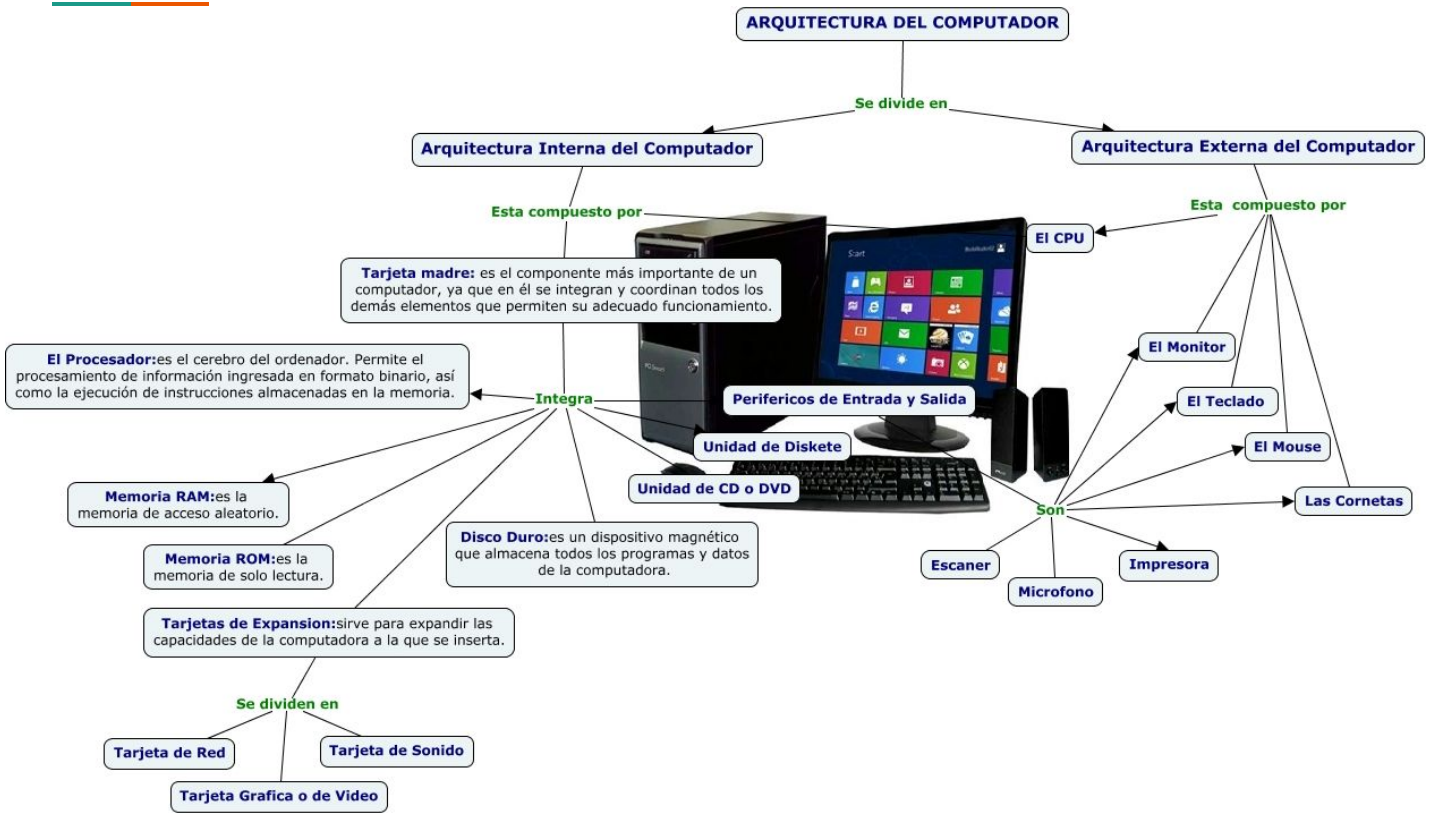
Unidad 1: Introducción a los Sistemas Operativos

¿Qué son los Sistemas Operativos de una computadora?

¿Qué tipos conocen?

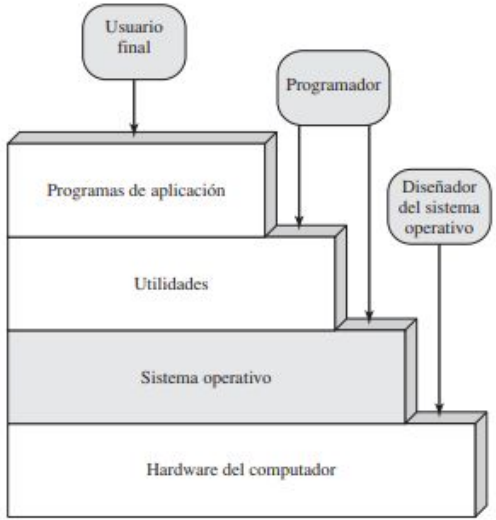
Sistemas Operativos modernos: versiones

Arquitectura: Componentes básicos



El sistema operativo como una interfaz de usuario/computador

El hardware y software utilizados para proporcionar aplicaciones a los usuarios se pueden ver de forma jerárquica o en capas, tal y como se muestra en la figura. El usuario de dichas aplicaciones, es decir, el usuario final, normalmente no se preocupa por los detalles del hardware del computador. Por tanto, el usuario final ve un sistema de computación en términos de un conjunto de aplicaciones. Una aplicación se puede expresar en un lenguaje de programación y normalmente es desarrollada por un programador de aplicaciones. Si un programador tuviera que desarrollar una aplicación como un conjunto de instrucciones en código máquina que se encargaran de controlar completamente el hardware del computador, se enfrentaría a una labor extremadamente compleja. Para facilitar esta tarea, se proporcionan un conjunto de programas de sistema.



El sistema operativo como gestor de recursos

Un computador es un conjunto de recursos que se utilizan para el transporte, almacenamiento y procesamiento de los datos, así como para llevar a cabo el control de estas funciones. El sistema operativo se encarga de gestionar estos recursos

¿Se puede decir que es el sistema operativo quien controla el transporte, almacenamiento y procesamiento de los datos? Desde un punto de vista, la respuesta es afirmativa: gestionando los recursos del computador, el sistema operativo tiene el control de las funciones básicas del mismo. Pero este control se realiza de una forma curiosa. Normalmente, se habla de un mecanismo de control como algo externo al dispositivo controlado pero no es el caso del sistema operativo, que es un mecanismo de control inusual en dos aspectos:

- Las funciones del sistema operativo actúan de la misma forma que el resto del software; es decir, se trata de un programa o conjunto de programas ejecutados por el procesador

- El sistema operativo frecuentemente cede el control y depende del procesador para volver a retomarlo.

El sistema operativo como gestor de recursos

Un sistema operativo es un conjunto de programas. Como otros programas, proporciona instrucciones para el procesador. La principal diferencia radica en el objetivo del programa. El sistema operativo dirige al procesador en el uso de los otros recursos del sistema y en la temporización de la ejecución de otros programas. No obstante, para que el procesador pueda realizar esto, el sistema operativo debe dejar paso a la ejecución de otros programas. Por tanto, el sistema operativo deja el control para que el procesador pueda realizar trabajo «útil» y de nuevo retoma el control para permitir al procesador que realice la siguiente pieza de trabajo

Servicios que proporciona un sistema operativo

- **Ejecución de programas:** Se necesita realizar una serie de pasos para ejecutar un programa. Las instrucciones y los datos se deben cargar en memoria principal. Los dispositivos de E/S y los ficheros se deben inicializar, y otros recursos deben prepararse. Los sistemas operativos realizan estas labores de planificación en nombre del usuario.
- **Acceso a dispositivos de E/S:** Cada dispositivo de E/S requiere su propio conjunto peculiar de instrucciones o señales de control para cada operación. El sistema operativo proporciona una interfaz uniforme que esconde esos detalles de forma que los programadores puedan acceder a dichos dispositivos utilizando lecturas y escrituras sencillas.
- **Acceso controlado a los ficheros:** Para el acceso a los ficheros, el sistema operativo debe reflejar una comprensión detallada no sólo de la naturaleza del dispositivo de E/S, sino también de la estructura de los datos contenidos en los ficheros del sistema de almacenamiento. Adicionalmente, en el caso de un sistema con múltiples usuarios, el sistema operativo puede proporcionar mecanismos de protección para controlar el acceso a los ficheros.

Servicios que proporciona un sistema operativo

- **Acceso al sistema**: Para sistemas compartidos o públicos, el sistema operativo controla el acceso al sistema completo y a recursos del sistema específicos. La función de acceso debe proporcionar protección a los recursos y a los datos, evitando el uso no autorizado de los usuarios y resolviendo conflictos en el caso de conflicto de recursos.
- **Detección y respuesta a errores:** Se pueden dar gran variedad de errores durante la ejecución de un sistema de computación. Éstos incluyen errores de hardware internos y externos, tales como un error de memoria, o un fallo en un dispositivo; y diferentes errores software, tales como la división por cero, el intento de acceder a una posición de memoria prohibida o la incapacidad del sistema operativo para conceder la solicitud de una aplicación. En cada caso, el sistema operativo debe proporcionar una respuesta que elimine la condición de error, suponiendo el menor impacto en las aplicaciones que están en ejecución. La respuesta puede oscilar entre finalizar el programa que causó el error hasta reintentar la operación o simplemente informar del error a la aplicación
- **Contabilidad**: Un buen sistema operativo recogerá estadísticas de uso de los diferentes recursos y monitorizará parámetros de rendimiento tales como el tiempo de respuesta. En cualquier sistema, esta información es útil para anticipar las necesidades de mejoras futuras y para optimizar el sistema a fin de mejorar su rendimiento. En un sistema multiusuario, esta información se puede utilizar para facturar a los diferentes usuarios

Tipos de sistemas operativos

Sistemas operativos de servidores:

Dan servicio a varios usuarios a la vez a través de una red y les permiten compartir los recursos de hardware y de software. Los servidores pueden proporcionar servicio de mail, de archivos, web, base de datos, etc. Estan especialmente diseñados para trabajar de forma eficiente debido a que su función es proporcionar servicios que deben estar disponibles la mayor cantidad de tiempo, deben ser ligeros y fáciles de administrar

Sistemas operativos de multiprocesadores:

Una manera cada vez más común de obtener más poder de cómputo es conectar varias CPU en un solo sistema. Dependiendo de la exactitud con la que se conecten y de lo que se comparta, estos sistemas se conocen como computadoras en paralelo, multicomputadoras o multiprocesadores. Necesitan sistemas operativos especiales, pero a menudo son variaciones de los sistemas operativos de servidores con características especiales para la comunicación, conectividad y consistencia.

Con la llegada de los chips multinúcleo para las computadoras personales, hasta los sistemas operativos de equipos de escritorio y portátiles convencionales empezaron a trabajar con multiprocesadores

Tipos de sistemas operativos

Sistemas operativos de computadoras personales :

Todos los sistemas operativos modernos soportan la multiprogramación, con frecuencia se inician docenas de programas al momento de arrancar el sistema. Su trabajo es proporcionar buen soporte para un solo usuario. Se utilizan ampliamente para el procesamiento de texto, las hojas de cálculo y el acceso a Internet

Sistemas operativos de dispositivos móviles:

Un Sistema operativo móvil o SO móvil es un conjunto de programas de bajo nivel que permite la abstracción de las peculiaridades específico del teléfono móvil y, provee servicios a las aplicaciones móviles, que se ejecutan sobre él. Al igual que los PC que utilizan Windows, Linux o Mac OS, los dispositivos móviles tienen sus sistemas operativos como Android, iOS, entre otros. Los sistemas operativos móviles son mucho más simples y están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

Tipos de sistemas operativos

Sistemas operativos integrados :

Los sistemas integrados (embedded), que también se conocen como incrustados o embebidos, operan en las computadoras que controlan dispositivos que no se consideran generalmente como computadoras, ya que no aceptan software instalado por el usuario.

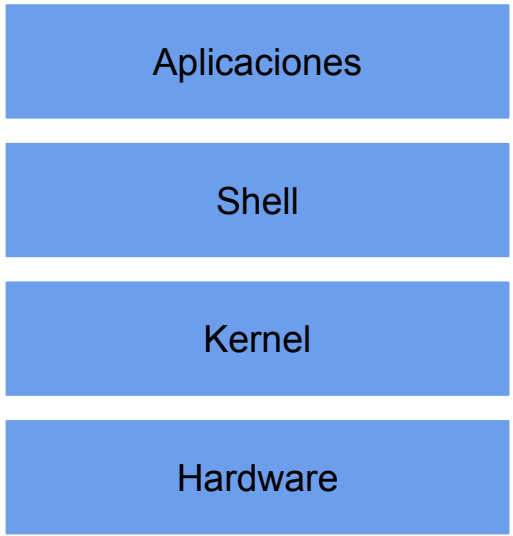
Sistemas operativos en tiempo real:

Estos sistemas se caracterizan por tener el tiempo como un parámetro clave. Si la acción debe ocurrir sin excepción en cierto momento (o dentro de cierto rango), tenemos un sistema en tiempo real duro. Muchos de estos sistemas se encuentran en el control de procesos industriales, en aeronáutica, en la milicia y en áreas de aplicación similares. Estos sistemas deben proveer garantías absolutas de que cierta acción ocurrirá en un instante determinado.

Sistemas operativos de tarjetas inteligentes

Los sistemas operativos más pequeños operan en las tarjetas inteligentes, que son dispositivos del tamaño de una tarjeta de crédito que contienen un chip de CPU. Tienen varias severas restricciones de poder de procesamiento y memoria. Algunas se energizan mediante contactos en el lector en el que se insertan, pero las tarjetas inteligentes sin contactos se energizan mediante inducción, lo cual limita en forma considerable las cosas que pueden hacer. Algunos sistemas de este tipo pueden realizar una sola función, como pagos electrónicos; otros pueden llevar a cabo varias funciones en la misma tarjeta inteligente.

Estructura básica de un sistema operativo



Aplicaciones de usuario

Una aplicación de usuario es un programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de tareas. (Ejemplos: MS Office - Word, Excel, Outlook, etc; Navegadores de Internet)

Shell

En la siguiente capa en comunicación directa con el kernel, tenemos el caparazón el shell. La shell actúa como un intérprete de comandos. En un sistema operativo existen distintos shell para recoger los comandos y acciones que introduce un usuario. De esta forma el sistema operativo realizará una serie de acciones en respuesta por ejemplo a un comando que escribamos en el terminal, o un clic de ratón.

Este programa siempre estará en funcionamiento en nuestro sistema registra todas las órdenes para decodificarlas y pasárselas al núcleo para realizar las acciones solicitadas. Prácticamente todas las órdenes que enviamos a la shell son programas ejecutables que están dentro del sistema de ficheros del sistema operativo.

Existen dos tipos de Shell y estos son:
Shells de texto común como *bash, emacs, símbolo del sistema de Windows, entre otros.*

Shells gráfico común como *GNome, KDE, XFCE, LXDE, Unity, MacOS Desktop Environment, Escritorio Windows, entre otros.*

Kernel

El kernel de un sistema es el componente central que sirve para dar vida al hardware. Es la capa responsable de asegurar que todos los programas y procesos tengan acceso a los recursos que necesitan de la máquina (memoria RAM, acceso a disco y el control de la CPU, por ejemplo) al mismo tiempo, de modo que haya un recurso compartido de estos. En otras palabras, es el cerebro del sistema operativo; el responsable de coordinar el acceso al hardware y los datos entre los diferentes componentes del sistema.

Unix vs Linux

Orígenes de Unix

Nació a principios de la década de los 70 por los desarrolladores Ken Thompson y Dennis Ritchie. Fue creado en los Laboratorios Bell, que pertenecen a la famosa compañía AT&T. Fue creado como un sistema operativo para manejar servidores, siendo un sistema operativo donde los comandos tienen casi todo el protagonismo.

Orígenes de Linux

El Kernel Linux fue creado por Linus Torvalds a principios de los 90. El Kernel se creó basándose en Unix y Linus tuvo la ayuda de otros grandes del software libre como Richard Stallman. A partir de ese año, se empezaron a crear muchas distribuciones basadas en Linux, así como muchos escritorios.

Unix vs Linux

Unix

Unix es un sistema propietario que no se puede modificar, propiedad de la compañía AT&T que es la única que tiene permiso de modificarlo y actualizarlo.

Linux

Como todos sabemos, Linux está bajo licencia GNU y por lo tanto, el kernel Linux es completamente libre y gratuito y cualquier persona puede modificar el código fuente, el cual está disponible para todo el mundo.

Procesos

El concepto más importante en cualquier sistema operativo es el de proceso, una abstracción de un programa en ejecución; todo lo demás depende de este concepto, por lo cual es importante que el diseñador del sistema operativo (y el estudiante) tenga una comprensión profunda acerca de lo que es un proceso lo más pronto posible.

Los procesos son una de las abstracciones más antiguas e importantes que proporcionan los sistemas operativos: proporcionan la capacidad de operar (pseudo) concurrentemente, incluso cuando hay sólo una CPU disponible. Convierten una CPU en varias CPU virtuales. Sin la abstracción de los procesos, la computación moderna no podría existir

Procesos

En cualquier sistema de multiprogramación, la CPU conmuta de un proceso a otro con rapidez, ejecutando cada uno durante décimas o centésimas de milisegundos: hablando en sentido estricto, en cualquier instante la CPU está ejecutando sólo un proceso, y en el transcurso de 1 segundo podría trabajar en varios de ellos, dando la apariencia de un paralelismo (o pseudoparalelismo, para distinguirlo del verdadero paralelismo de hardware de los sistemas multiprocesadores con dos o más CPUs que comparten la misma memoria física). Es difícil para las personas llevar la cuenta de varias actividades en paralelo; por lo tanto, los diseñadores de sistemas operativos han evolucionado con el paso de los años a un modelo conceptual (procesos secuenciales) que facilita el trabajo con el paralelismo.

El modelo conceptual de los procesos secuenciales

En este modelo, todo el software ejecutable en la computadora, que algunas veces incluye al sistema operativo, se organiza en varios procesos secuenciales (procesos, para abreviar). Un proceso no es más que una instancia de un programa en ejecución, incluyendo los valores actuales del contador de programa, los registros y las variables. En concepto, cada proceso tiene su propia CPU virtual; en la realidad, la CPU real conmuta de un proceso a otro, pero para entender el sistema es mucho más fácil pensar en una colección de procesos que se ejecutan en (pseudo) paralelo, en vez de tratar de llevar la cuenta de cómo la CPU conmuta de programa en programa. Esta conmutación rápida de un proceso a otro se conoce como multiprogramación

Creación de un proceso

Los sistemas operativos necesitan cierta manera de crear procesos. En sistemas muy simples o sistemas diseñados para ejecutar sólo una aplicación (por ejemplo, el controlador en un horno de microondas), es posible tener presentes todos los procesos que se vayan a requerir cuando el sistema inicie. No obstante, en los sistemas de propósito general se necesita cierta forma de crear y terminar procesos según sea necesario durante la operación. Ahora analizaremos varias de estas cuestiones

Hay cuatro eventos principales que provocan la creación de procesos:

1. El arranque del sistema.
2. La ejecución, desde un proceso, de una llamada al sistema para creación de procesos.
3. Una petición de usuario para crear un proceso.
4. El inicio de un trabajo por lotes.

Creación de un proceso

Además de los procesos que se crean al arranque, posteriormente se pueden crear otros. A menudo, un proceso en ejecución emitirá llamadas al sistema para crear uno o más procesos nuevos, para que le ayuden a realizar su trabajo. En especial, es útil crear procesos cuando el trabajo a realizar se puede formular fácilmente en términos de varios procesos interactivos relacionados entre sí, pero independientes en los demás aspectos. Por ejemplo, si se va a obtener una gran cantidad de datos a través de una red para su posterior procesamiento, puede ser conveniente crear un proceso para obtener los datos y colocarlos en un búfer compartido, mientras un segundo proceso remueve los elementos de datos y los procesa. En un multiprocesador, al permitir que cada proceso se ejecute en una CPU distinta también se puede hacer que el trabajo se realice con mayor rapidez

Creación de un proceso

En los sistemas interactivos, los usuarios pueden iniciar un programa escribiendo un comando o haciendo (doble) clic en un icono. Cualquiera de las dos acciones inicia un proceso y ejecuta el programa seleccionado. En los sistemas UNIX basados en comandos que ejecutan X, el nuevo proceso se hace cargo de la ventana en la que se inició. En Microsoft Windows, cuando se inicia un proceso no tiene una ventana, pero puede crear una (o más) y la mayoría lo hace. En ambos sistemas, los usuarios pueden tener varias ventanas abiertas a la vez, cada una ejecutando algún proceso.

Creación de un proceso

La última situación en la que se crean los procesos se aplica sólo a los sistemas de procesamiento por lotes que se encuentran en las mainframes grandes. Aquí los usuarios pueden enviar trabajos de procesamiento por lotes al sistema (posiblemente en forma remota). Cuando el sistema operativo decide que tiene los recursos para ejecutar otro trabajo, crea un proceso y ejecuta el siguiente trabajo de la cola de entrada

.

Creación de un proceso

Técnicamente, en todos estos casos, para crear un proceso hay que hacer que un proceso existente ejecute una llamada al sistema de creación de proceso. Ese proceso puede ser un proceso de usuario en ejecución, un proceso del sistema invocado mediante el teclado o ratón, o un proceso del administrador de procesamiento por lotes. Lo que hace en todo caso es ejecutar una llamada al sistema para crear el proceso. Esta llamada al sistema indica al sistema operativo que cree un proceso y le indica, directa o indirectamente, cuál programa debe ejecutarlo

.

Terminación de procesos

Una vez que se crea un proceso, empieza a ejecutarse y realiza el trabajo al que está destinado. Sin embargo, nada dura para siempre, ni siquiera los procesos. Tarde o temprano el nuevo proceso terminará, por lo general debido a una de las siguientes condiciones:

1. Salida normal (voluntaria).
2. Salida por error (voluntaria).
3. Error fatal (involuntaria).
4. Eliminado por otro proceso (involuntaria).

.

Terminación de procesos

La mayoría de los procesos terminan debido a que han concluido su trabajo. Cuando un compilador ha compilado el programa que recibe, ejecuta una llamada al sistema para indicar al sistema operativo que ha terminado. Esta llamada es exit en UNIX y ExitProcess en Windows. Los programas orientados a pantalla también admiten la terminación voluntaria.

.

Terminación de procesos

La segunda razón de terminación es que el proceso descubra un error. Por ejemplo, si un usuario escribe el comando para compilar el programa foo.c y no existe dicho archivo, el compilador simplemente termina.

Los procesos interactivos orientados a pantalla por lo general no terminan cuando reciben parámetros incorrectos. En vez de ello, aparece un cuadro de diálogo y se le pide al usuario que intente de nuevo.

.

Terminación de procesos

La tercera razón de terminación es un error fatal producido por el proceso, a menudo debido a un error en el programa. Algunos ejemplos incluyen el ejecutar una instrucción ilegal, hacer referencia a una parte de memoria no existente o la división entre cero. En algunos sistemas (como UNIX), un proceso puede indicar al sistema operativo que desea manejar ciertos errores por sí mismo, en cuyo caso el proceso recibe una señal (se interrumpe) en vez de terminar.

.

Terminación de procesos

La cuarta razón por la que un proceso podría terminar es que ejecute una llamada al sistema que indique al sistema operativo que elimine otros procesos. En UNIX esta llamada es kill. La función correspondiente en Win32 es TerminateProcess. En ambos casos, el proceso eliminador debe tener la autorización necesaria para realizar la eliminación. En algunos sistemas, cuando un proceso termina (ya sea en forma voluntaria o forzosa) todos los procesos que creó se eliminan de inmediato también.

Estados de un proceso

Aunque cada proceso es una entidad independiente, con su propio contador de programa y estado interno, a menudo los procesos necesitan interactuar con otros. Un proceso puede generar cierta salida que otro proceso utiliza como entrada. En el comando de shell:

cat capitulo1 capitulo2 capitulo3 | grep arbol

el primer proceso, que ejecuta cat, concatena tres archivos y los envía como salida. El segundo proceso, que ejecuta grep, selecciona todas las líneas que contengan la palabra “arbol”. Dependiendo de la velocidad relativa de los dos procesos (que dependen tanto de la complejidad relativa de los programas, como de cuánto tiempo ha tenido cada uno la CPU), puede ocurrir que grep esté listo para ejecutarse, pero que no haya una entrada esperándolo. Entonces debe bloquear hasta que haya una entrada disponible.

Estados de un proceso

Cuando un proceso se bloquea, lo hace debido a que por lógica no puede continuar, comúnmente porque está esperando una entrada que todavía no está disponible. También es posible que un proceso, que esté listo en concepto y pueda ejecutarse, se detenga debido a que el sistema operativo ha decidido asignar la CPU a otro proceso por cierto tiempo. Estas dos condiciones son completamente distintas. En el primer caso, la suspensión está inherente en el problema (no se puede procesar la línea de comandos del usuario sino hasta que éste la haya escrito mediante el teclado). En el segundo caso, es un tecnicismo del sistema (no hay suficientes CPUs como para otorgar a cada proceso su propio procesador privado).

Estados de un proceso

1. El proceso se bloquea para recibir entrada
2. El planificador selecciona otro proceso
3. El planificador selecciona este proceso
4. La entrada ya está disponible

En la figura podemos ver un diagrama de estados que muestra los tres estados en los que se puede encontrar un proceso:

1. En ejecución (en realidad está usando la CPU en ese instante).
2. Listo (ejecutable; se detuvo temporalmente para dejar que se ejecute otro proceso).
3. Bloqueado (no puede ejecutarse sino hasta que ocurra cierto evento externo).

En sentido lógico, los primeros dos estados son similares. En ambos casos el proceso está deseoso de ejecutarse; sólo en el segundo no hay temporalmente una CPU para él. El tercer estado es distinto de los primeros dos en cuanto a que el proceso no se puede ejecutar, incluso aunque la CPU no tenga nada que hacer.