UNIVERSIDAD DE CARABOBO
Facultad Experimental de Ciencias y Tecnología
Departamento de Computación

Documento Vertical:
Administración de Dispositivos

Profesora:
Mirella Herrera

Alumnos
Acosta S, María Gabriela
Gomez Carlos
Gomez Leosbeth
Rodriguez Jose
Rodriguez Jissel

Naguanagua, Octubre de 2014

Administrador de entrada/salida

El sistema de entrada y salida es la parte del S.O. encargada de la administración de los dispositivos de E/S ofreciendo un visión lógica y simplificada de los mismos para que pueda ser usada por otros componentes del sistema operativo (como el sistema de archivos) o incluso por el usuario. Mediante esta visión lógica se ofrece a los usuarios un mecanismo de abstracción que oculta todos los detalles relacionados con los dispositivos físicos, así como el funcionamiento real de los mismos.

El SO debe controlar el funcionamiento de todos los dispositivos de E/S para alcanzar los siguientes objetivos:

- Facilitar el manejo de los dispositivos periféricos. Para ello debe ofrecer una interfaz entre los dispositivos y el resto del sistema que sea sencilla y fácil de utilizar.
- Optimizar la E/S del sistema, proporcionando mecanismos de incremento de prestaciones donde sea necesario.
- Proporcionar dispositivos virtuales que permitan conectar cualquier tipo de dispositivos físicos sin que sea necesario remodelar el sistema de E/S del SO.
- Permitir la conexión de dispositivos nuevos de E/S, solventando de forma automática su instalación usando mecanismos del tipo plug & play.

Arquitectura del sistema de entrada/salida

El sistema de entrada/salida está construido como un conjunto de manejadores apilados, cada uno de los cuales está asociado a un dispositivo de entrada/salida (archivos, red, etc.). Ofrece a las aplicaciones y entornos de ejecución servicios genéricos que permiten manejar los objetos de entrada/salida del sistema. A través de ellos se puede acceder a todos los manejadores de archivos y de dispositivos tales como discos, cintas, redes, consola, tarjetas de sonido, etc.

La arquitectura del sistema de entrada/salida es compleja y está estructurada en capas, cada una de las cuales tiene una funcionalidad bien definida:

- **Interfaz del sistema operativo para entrada/salida.** Proporciona servicios de E/S síncrona y asíncrona a las aplicaciones y una interfaz homogénea para poderse comunicar con los manejadores de dispositivo ocultando los detalles de bajo nivel.

- **Sistemas de archivos.** Proporcionan una interfaz homogénea, a través del sistema de archivos virtuales, para acceder a todos los sistemas de archivos que proporciona el sistema operativo (FFS, SV, NTFS, FAT, etc.). Permite acceder a los manejadores de los dispositivos de almacenamiento de forma transparente, incluyendo en muchos casos, como NFS o NTFS, accesos remotos a través de redes. En algunos sistemas, como Windows NT, los servidores para cada tipo de sistema de archivos se pueden cargar y descargar dinámica- mente como cualquier otro manejador.

- **Gestor de redes.** Proporciona una interfaz homogénea para acceder a todos los sistemas de red que proporciona el sistema operativo (TCP/IP, Novell, etc.). Además, permite acceder a los manejadores de cada tipo de red particular de forma transparente.

- **Gestor de bloques**. Los sistemas de archivos y otros dispositivos lógicos con acceso a nivel de bloque se suelen limitar a traducir las operaciones del formato del usuario al de bloques que entiende el dispositivo y se las pasan a este gestor de bloques. El gestor de bloques admite únicamente operaciones a nivel de bloque e interacciona con la cache de bloques para optimizar la E/S.

- **Gestor de cache.** Optimiza la entrada/salida mediante la gestión de almacenamiento intermedio en memoria para dispositivos de E/S de tipo bloque. Es importante saber que su tamaño varía dinámicamente en función de la memoria RAM disponible y que los bloques se escriben a los dispositivos según una política bien definida, que en UNIX y WINDOWS NT es la de escritura retrasada.

- **Manejadores de dispositivo.** Proporcionan operaciones de alto nivel sobre los dispositivos y las traducen en su ámbito interno a operaciones de control de cada dispositivo particular. Como ya se ha dicho, los manejadores se comunican con los dispositivos reales mediante puertos o zonas de memoria especiales.

Conexión de un dispositivo de E/S a una computadora

En el modelo de un periférico se distinguen dos elementos:

- **Periféricos o dispositivos de E/S.** Elementos que se conectan a la unidad central de proceso a través de las unidades de entrada/salida. Son el componente mecánico que se conecta a la computadora.

- **Controladores de dispositivos o unidades de E/S.** Se encargan de hacer la transferencia de información entre la memoria principal y los periféricos. Son el componente electrónico a través del cual se conecta el dispositivo de

E/S. Tienen una conexión al bus de la computa dora y otra para el dispositivo (generalmente mediante cables internos o externos).

Los controladores son muy variados, casi tanto como los dispositivos de E/S. Muchos de ellos, como los de disco, pueden controlar múltiples dispositivos. Otros, como los de canales de E/S, incluyen su propia CPU y bus para controlar la E/S por programa y evitar interrupciones en la UCP de la computadora. De cualquier forma, en los últimos años ha existido un esfuerzo importante de estandarización de los dispositivos, lo que permite usar un mismo controlador para dispositivos de distintos fabricantes. Un buen ejemplo lo constituyen los dispositivos SCSI (SrnaII Computer Svstern interftice), cuyos controladores ofrecen una interfaz común independientemente de que se trate de un disco, una cinta, un CD-ROM, etc.

El controlador es el componente más importante desde el punto de vista del sistema operativo ya que constituye la interfaz del dispositivo con el bus de la computadora y es el componente que se ve desde la CPU. Hay tres registros importantes en casi todos los controladores: **registro de datos, estado y control.** El registro de datos sirve para el intercambio de datos. En él irá el controlador cargando los datos leídos y de él irá extrayendo los datos para su escritura en el periférico. Un bit del registro de estado sirve para indicar que el controlador puede transferir una palabra. En las operaciones de lectura esto significa que ha cargado en el registro de datos un nuevo valor, mientras que en las de escritura significa que necesita un nuevo dato. Otros bits de este registro sirven para que el controlador indique los problemas que ha encontrado en la ejecución de la última operación de E/S. El registro de control sirve para indicarle al controlador las operaciones que ha de realizar. Los distintos bits de este registro indican distintas acciones que ha de realizar el periférico. Para empezar una operación de E/S, la UCP tiene que escribir sobre los registros anteriores los datos de la operación a través de una dirección de E/S o de memoria asignada únicamente al controlador.
Las características del controlador son muy importantes, ya que definen el aspecto del periférico para el sistema operativo. Atendiendo a las características del hardware de los dispositivos, se pueden observar los siguientes aspectos distintivos:

- **Dirección de E/S.** En general hay dos modelos de direccionamiento de E/S, los que usan puertos y los que proyectan los registros en memoria.

- **Unidad de transferencia**. Los dispositivos suelen usar unidades de transferencia de tamaño fijo. Hay dos modelos clásicos de dispositivos: de caracteres y de bloques.

- **Interacción computadora-controlador.** La computadora tiene que interaccionar con la computadora para realizar las operaciones de E/S y saber cuándo terminan.

Organización de las funciones de E/S

Tres técnicas para realizar la E/S:

- E/S programada: El procesador emite una orden de E/S de parte de un proceso a un módulo de E/S; el proceso espera entonces a que termine la operación, antes de seguir.

- E/S dirigida por interrupciones: El procesador emite una orden de E/S de parte de un proceso, continúa la ejecución de las instrucciones siguientes y es interrumpido por el módulo de E/S cuando este ha completado su trabajo. Las instrucciones siguientes pueden ser del mismo proceso, si no es necesario para este esperar la terminación de la E/S. En otro caso, el proceso se ve suspendido a la espera de la interrupción, mientras se realiza otro trabajo.

- E/S por DMA: Un módulo de DMA controla el intercambio de datos entre la memoria principal y un módulo de E/S. El procesador envía una petición de transferencia de un bloque de datos al modulo de DMA y se interrumpe solo cuando se ha transferido el bloque entero.

Dispositivos de entrada y salida

Dispositivos o periféricos son aparatos solidos e independientes que van conectados a la unidad central de la computadora. Existen diversos tipos de dispositivos con funciones diferentes.
Estos se dividen en tres grandes grupos:

1. **Dispositivos de interfaz de usuario:** Permiten la comunicación usuario-computadora, los cuales se dividen en:
   - **Dispositivos de entrada:** tienen la tarea de introducir información, órdenes, comandos e instrucciones para que la computadora lo procese con el sistema operativo. Los datos se leen de estos dispositivos y se almacenan en la memoria central o interna. Entre los más conocidos se encuentran:
     - El Teclado: Es el principal dispositivo para introducir órdenes e información al sistema. Un teclado alfanumérico se utiliza principalmente como un dispositivo para introducir texto. El teclado es un dispositivo eficaz para introducir datos no gráficos como rótulos de imágenes asociados con un despliegue de gráficas. Los teclados también pueden ofrecerse con características que facilitan la entrada de coordenadas de la pantalla, selecciones de menús o funciones de gráficas.
     - El Mouse: Dispositivo fundamental que sirve para señalar y accionar los diversos elementos de la pantalla del computador.
     - El Escáner: Este dispositivo nos permite convertir en formato digital los textos, fotografías e imágenes impresas.

   - **Dispositivos de salida:** son los que se encargan de procesar los resultados de salida de los procesos de datos. Y mostrarlos al usuario.

Entre ellos tenemos:

- El Monitor: Es el principal dispositivos en procesar los datos y mostrárselos al usuario.
- La Impresora: Es el periférico encargado de hacer copias en papel de la información que nos muestra el computador.

2. **Dispositivos de almacenamiento:** Proporcionan almacenamiento no volátil de datos y memoria. Entren los dispositivos almacenamiento se encuentran:
    - **Unidad de lectora de CD –ROM**: Permite leer datos, los cuales se graban en el disco duro o pueden ser leídos para obtener información.
    - **Disco Duro:** Proporcionan un acceso más rápido a los datos que los discos flexibles y pueden almacenar mucha más información.

3. **Dispositivos de comunicaciones:** Permiten la conexión de la computadora con otras computadoras. Ej. Módems, tarjetas , etc

Operaciones de entrada/salida

Las funciones que debe realizar un computador para ejecutar trabajos de entrada/salida:

- Direccionamiento o selección del dispositivo que debe llevar a cabo la operación de E/S.
- Transferencia de los datos entre el procesador y el dispositivo (en uno u otro sentido).
- Sincronización y coordinación de las operaciones.

Esta última función es necesaria debido a la deferencia de velocidades entre los dispositivos y la CPU y a la independencia que debe existir entre los periféricos y la CPU (por ejemplo, suelen tener relojes diferentes).
Se define una transferencia elemental de información como la transmisión de una sola unidad de información (normalmente un byte) entre el procesador y el periférico o viceversa. Para efectuar una transferencia elemental de información son precisas las siguientes funciones:

- Establecimiento de una comunicación física entre el procesador y el periférico para la transmisión de la unidad de información.
- Control de los periféricos, en que se incluyen operaciones como prueba y modificación del estado del periférico. Para realizar estas funciones la CPU gestionara las líneas de control necesarias.

Definiremos una operación de E/S como el conjunto de acciones necesarias para la transferencia de un conjunto de datos (es decir, una transferencia completa de datos). Para la realización de una operación de E/S se deben efectuar las siguientes funciones:

- -Recuento de las unidades de información transferidas (normalmente bytes) para reconocer el fin de operación.

- -Sincronización de velocidad entre la CPU y el periférico.
- Detección de errores (e incluso corrección) mediante la utilización de los códigos necesarios (bits de paridad, códigos de redundancia cíclica, etc.)
- Almacenamiento temporal de la información. Es más eficiente utilizar un buffer temporal específico para las operaciones de E/S que utilizan el área de datos del programa.
- Conversión de códigos, conversión serie/paralelo, etc.

Para las operaciones de E/S son posibles las tres técnicas siguientes:

➢ **E/S programada** : Cuando el procesador está ejecutando un programa y encuentra una instrucción de E/S, ejecuta dicha instrucción, enviando una orden al módulo apropiado de E/S. Con E/S programada, el módulo de E/S llevará a cabo la acción requerida y luego activará los bits apropiados en el registro de estado de E/S. El módulo de E/S no lleva a cabo ninguna otra acción para avisar al procesador. En particular, no interrumpe al procesador. Así pues, es responsabilidad del procesador comprobar periódicamente el estado del módulo de E/S hasta saber que se ha completado la operación. Con esta técnica, el procesador es el responsable de extraer los datos de la memoria principal cuando va a hacer una salida o poner los datos en la memoria principal cuando se hace una entrada. El software de E/S se escribe de manera tal que el procesador ejecute unas instrucciones que le otorguen el control directo sobre la operación de E/S, incluyendo la comprobación del estado de los dispositivos, el envío de órdenes de lectura o escritura y la transferencia de los datos. Por lo tanto, en el conjunto de instrucciones se incluyen instrucciones de E/S de las siguientes categorías:

• Control: Empleadas para activar un dispositivo externo y decirle qué debe hacer. Por ejemplo, una unidad de cinta magnética puede ser instruida para rebobinar o avanzar un registro.

• Comprobación: Empleadas para comprobar varias condiciones de estado asociadas con un módulo de E/S y sus periféricos.

• Lectura, escritura: Empleadas para transferir los datos entre los registros del procesa- dor y los dispositivos externos.

El diagrama que se muestra en la figura 1.21 destaca la desventaja principal de esta técnica: Es un proceso que consume tiempo y que mantiene al procesador ocupado de forma innecesaria.

Software de entrada salida

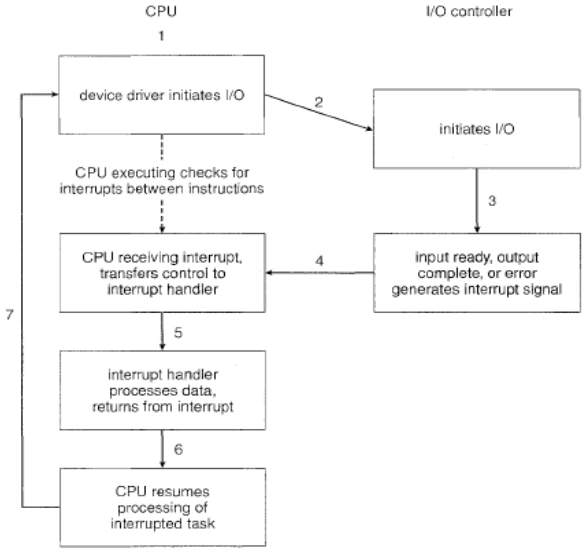
Este software se organiza en una serie de capas. Estas capas se corresponden, en general, con los niveles de la arquitectura de E/S.

Cuando un proceso solicita una operación de E/S, el sistema operativo prepara dicha operación y bloquea al proceso hasta que se recibe una interrupción del controlador del dispositivo indicando que la operación está completa. Las peticiones se procesan de forma estructurada en las siguientes capas:

- Manejador de interrupción.
- Manejadores de dispositivos o drivers.
- Software de E/S independiente de los dispositivos. Este software está formado por la parte de alto nivel de los manejadores, el gestor de cache, el gestor de bloques y el servidor de archivos.
- Interfaz del sistema operativo. Llamadas al sistema que usan las aplicaciones de usuario.

Manejador de interrupción

Los manejadores de interrupción se encargan de tratar las interrupciones que generan los controladores de dispositivos una vez que éstos están listos para la transferencia de datos o bien han leído o escrito los datos de memoria principal en caso de acceso directo a memoria. Para tratar dicha interrupción se ejecuta el correspondiente manejador de interrupción cuyo efecto es el de salvar los registros, comunicar el evento al manejador del dispositivo y restaurar la ejecución de un proceso (que no tiene por qué ser el interrumpido).



Ciclo de entrada salida dirigido por interrupción

Manejador de dispositivos de E/S o drivers.

Cada dispositivo de E/S, o cada clase de dispositivos, tiene un manejador asociado en el sistema operativo. Dicho manejador incluye: código independiente del dispositivo para proporcionar al nivel superior del sistema operativo una interfaz de alto nivel y el código dependiente del dispositivo necesario para programar el controlador del dispositivo a través de sus registros y mandatos. La tarea de un manejador de dispositivo es aceptar peticiones en formato abstracto, de la parte del código de E/S independiente del dispositivo, traducir dichas peticiones a términos que entienda el controlador, enviar al mismo las órdenes adecuadas en la secuencia correcta y esperar a que se cumplan.

Todos los manejadores tienen una lista de peticiones pendientes por dispositivo donde se encolan las peticiones que llegan de niveles superiores. El manejador explora la lista de peticiones, extrae una petición pendiente y ordena su ejecución. La política de extracción de peticiones de la lista es dependiente de manejador y puede ser FIFO u otro tipo política. Una vez enviada la petición al controlador, el manejador se bloquea o no, dependiendo de la velocidad del dispositivo. Para los lentos (discos) se bloquea esperando una interrupción. Para los rápidos (pantalla, discos RAM, etcétera) responde inmediatamente. Después de recibir el fin de operación, controla la existencia de errores y devuelve al nivel superior el estado de terminación de la operación. Si tiene operaciones pendientes en la cola de peticiones, atiende a la siguiente, en caso de que le toque ejecutar después de la operación de E/S. En caso contrario se bloquea.

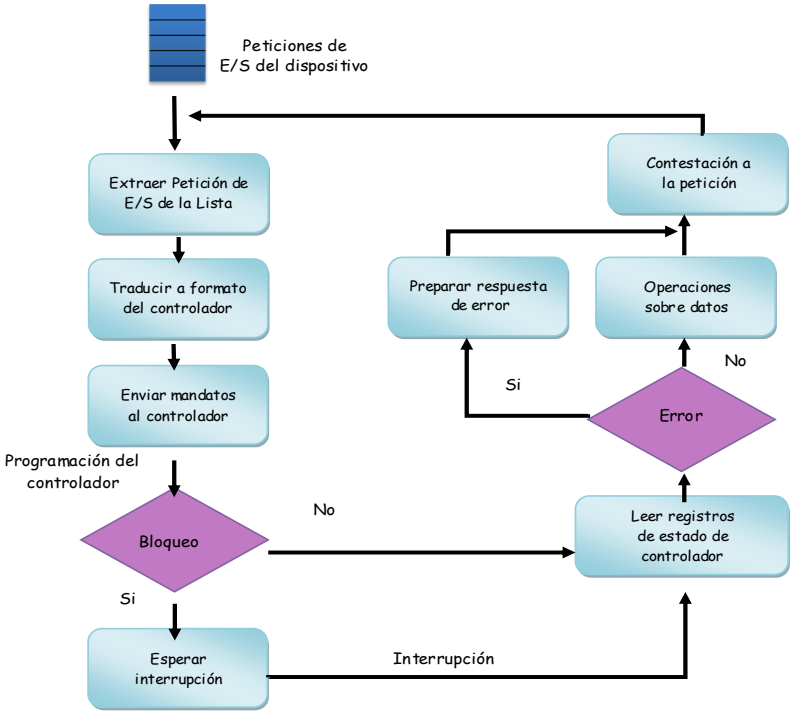


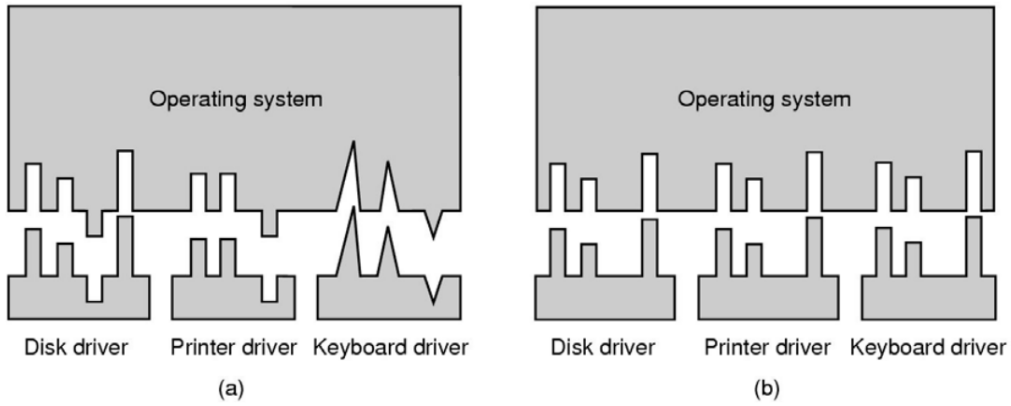
Diagrama de flujo con las operaciones de un manejador

Software de E/S independiente del dispositivo

La mayor parte del sistema de E/S es software independiente de dispositivo. Este nivel incluye el sistema de archivos y el de gestión de red, el gestor de bloques, la cache de bloques y una parte de los manejadores de dispositivo. La principal función de esta capa de software es ejecutar las funciones de E/S que son comunes a todos los dispositivos a través de una interfaz uniforme, Internamente, en este nivel se proporciona acceso a nivel de bloques o caracteres, almacenamiento intermedio, gestión de los dispositivos, planificación de la E/S y control de errores.

Un aspecto de esta cuestión es la interfaz entre los drivers de dispositivo y el resto del sistema operativo. En la Figura (a) de abajo se ilustra una situación en la

que cada driver de dispositivo tiene una interfaz diferente con el sistema operativo, lo que significa que las funciones del driver que el sistema puede invocar difieren de un driver a otro. También podría significar que las funciones del núcleo que necesita el driver difieren también de un controlador a otro. En conjunto, todo esto significa que la interfaz con cada nuevo driver requiere un elevado nuevo esfuerzo de programación.



(a)Sin una interfaz estándar con los drivers (b) Con una interfaz estándar con los drivers

El tamaño de acceso a nivel de bloques se hace usando tamaños de bloque de acceso comunes para todo un sistema de archivos, lo que permite ocultar que cada dispositivo puede tener distinto tamaño de sector y distinta geometría.

En contraste, en la Figura (b) se muestra un diseño diferente en el cual todos los drivers tienen la misma interfaz. En este caso resulta mucho más fácil añadir un nuevo driver, siempre y cuando se ajuste a la interfaz con los drivers existente. Además en este caso los escritores de drivers conocen perfectamente lo que se espera de ellos (es decir, qué funciones debe proporcionar el driver y qué funciones del núcleo puede invocar el driver). En la práctica, no todos los dispositivos son absolutamente idénticos, pero usualmente sólo hay un pequeño número de tipos de dispositivos e incluso esos tipos son generalmente casi el mismo. Por ejemplo, incluso los dispositivos de bloques y de caracteres tienen muchas funciones en común.

Existe un almacenamiento intermedio para optimizar la E/S y para armonizar las peticiones de usuarios, que pueden ser de cualquier tamaño, con bloques que maneja el sistema de archivos. Esto se usa para tres cosas:

• Optimizar la E/S evitando acceso a los dispositivos.
• Ocultar las diferencias de velocidad con la que cada dispositivo y usuario manejan los datos.
• Facilitar la implementación de la semántica de compartición, al existir una única copia de los datos en memoria.

El sistema de E/S mantiene buffers en distintos componentes.

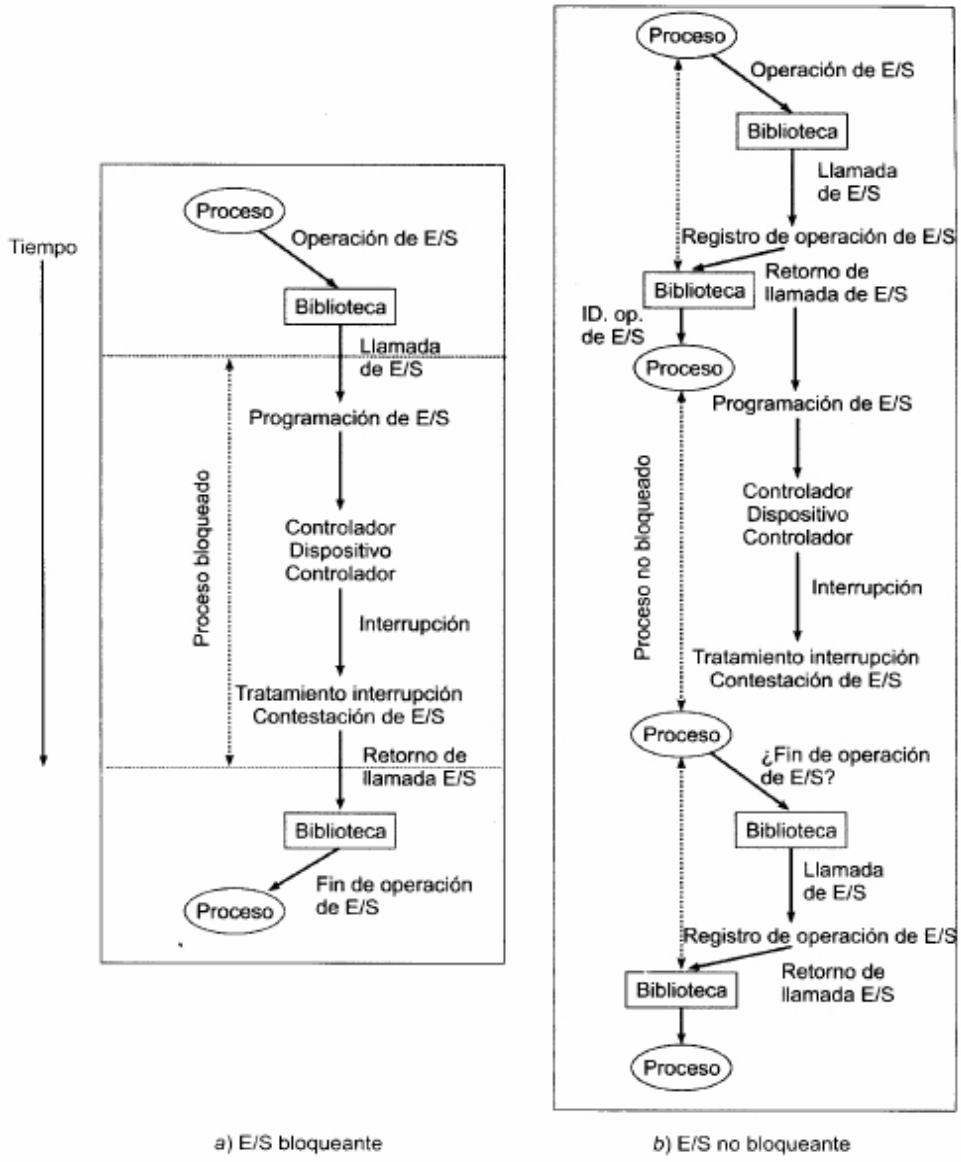
Una de las funciones principales del sistema de E/S es la planificación de la E/S de los distintos componentes. Para ello se usan colas de peticiones para cada clase de dispositivo, de las que se extraen las peticiones de cada dispositivo en particular. Cada una de éstas se ordena con una determinada política.

Por último, este nivel proporciona gestión de errores para aquellos casos que el manejador de dispositivo no puede solucionar. En general, todos los sistemas operativos incluyen alguna forma de control de errores internos y de notificación al exterior en caso de que esos errores no se puedan resolver.

E/S bloqueante y no bloqueante

La mayoría de los dispositivos de E/S son no bloqueantes, también llamados asíncronos, es decir, reciben la operación, la programan, contestan e interrumpen al cabo de un cierto tiempo. Sólo los dispositivos muy rápidos o algunos dedicados fuerzan la existencia de operaciones de E/S bloqueantes (también llamadas síncronas). Sin embargo, la mayoría de las aplicaciones efectúan operaciones de E/S con lógica bloqueante, lo que significa que emiten la operación y esperan hasta tener el resultado antes de continuar su ejecución. En este tipo de operaciones, el sistema operativo recibe la operación y bloquea al proceso emisor hasta que la operación de E/S ha terminado, momento en que desbloquea a la aplicación y le envía el estado del resultado de la operación. En este caso, la aplicación puede acceder a los datos inmediatamente, ya que los tiene disponibles en la posición de memoria especificada, a no ser que hubiera un error de E/S. Este modelo de programación es claro y sencillo, por lo que las principales llamadas al sistema de E/S. como read o write en POSIX y ReadFile y WriteFile en Win32, bloquean al usuario y completan la operación antes de devolver el control al usuario.

Las llamadas de E/S no bloqueantes se comportan de forma muy distinta, reflejando mejor la propia naturaleza del comportamiento de los dispositivos de E/S. Estas llamadas permiten a la aplicación seguir su ejecución, sin bloquearla, después de hacer una petición de E/S. El procesamiento de la llamada de E/S consiste en recuperar los parámetros de la misma, asignar un identificador de operación de E/S pendiente de ejecución y devolver a la aplicación este identificador.



Flujo de las operaciones de E/S bloqueantes y no bloqueantes

Almacenamiento intermedio

Supongamos que un proceso de usuario desea leer bloques de datos de una cinta, uno cada vez, siendo cada bloque de 100 bytes. Los datos van a ser leídos en una zona de datos del proceso de usuario situada en las direcciones virtuales 1000 a 1009. La forma más sencilla de hacerlo sería emitir una orden de E/S (parecida a "Leer Bloque[1000, cinta]") a la unidad de cinta y esperar a que los datos estén disponibles. La espera podría ser activa (comprobando continuamente el estado del dispositivo) o, de manera más práctica, suspender al proceso en espera de una interrupción.

Hay dos problemas con este enfoque. En primer lugar, el programa se queda colgado esperando a que la relativamente lenta operación de E/S termine. El segundo problema es que este método de E/S dificulta las decisiones de intercambio del sistema operativo. Las ubicaciones virtuales 1000 a 1009 deben permanecer en memoria principal durante el curso de la transferencia del bloque. De lo contrario, parte de los datos se perderán. Si se está utilizando paginación, la página que contenga dichas direcciones virtuales, por lo menos, debe permanecer en memoria principal. De este modo, aunque algunas partes del proceso puedan ser expulsadas a disco, es imposible expulsar al proceso por completo, aunque el sistema operativo lo desee. Nótese también que hay riesgo de interbloqueo de un solo proceso. Si un proceso emite una orden de E/S, queda suspendido a la espera del resultado, se le expulsa antes de comenzar la operación y se bloquea esperando a que la operación termine. Mientras tanto, la operación de E/S queda bloqueada esperando a que el proceso vuelva a memoria. Para evitar este interbloqueo, la memoria de usuario implicada en la operación de E/S debe quedar fija en la memoria principal, inmediatamente después de emitir la petición de E/S, incluso aunque la operación de E/S se encole y pueda no ejecutarse por algún tiempo.
Las mismas consideraciones pueden aplicarse a las operaciones de salida. Si se transfiere un bloque desde el área de un proceso de usuario hacia un módulo de E/S directamente, el proceso se bloqueará durante la transferencia y no puede ser expulsado.
Para evitar esta carga e incapacidad, a veces es conveniente llevar a cabo las transferencias de entrada por adelantado a las peticiones y realizar las transferencias de salida un tiempo después de hacer la petición. Esta técnica se conoce con el nombre de almacenamiento intermedio (buffering).
A la hora de discutir los distintos métodos de almacenamiento intermedio, es a veces importante hacer una distinción entre dos tipos de dispositivos: dispositivos de bloques y dispositivos de flujo. Los dispositivos de bloques almacenan la información en bloques, normalmente de un tamaño fijo, siendo las transferencias de un bloque cada vez. Generalmente, es posible referirse a los bloques por un número de bloque. Los discos y las cintas son ejemplos de dispositivos de bloques. Los dispositivos de flujo transfieren los datos como flujos de bytes; no poseen estructura de bloques. Los Terminales, impresoras, puertos de comunicación, ratones y otros dispositivos apuntadores y la mayoría de los dispositivos restantes que no son de almacenamiento secundario son dispositivos de flujo.

Buffer Sencillo

La clase de apoyo más simple que el sistema operativo puede ofrecer es el buffer sencillo. Cuando un proceso de usuario realiza una petición de E/S, el sistema operativo le asigna a la operación un buffer en la parte del sistema de la memoria principal.

Para los dispositivos de bloques, el esquema del buffer sencillo puede describirse como sigue. Las transferencias de entrada se realizan al buffer del sistema. Cuando se ha completado latransferencia, el proceso mueve el bloque al espacio del usuario y pide otro bloque inmediatamente. Esta técnica se llama lectura por adelantado o

entrada anticipada: se realiza esperando que el bloque se necesite más adelante. Para muchos tipos de operaciones, ésta suposición es razonable la mayoría de las veces. La lectura del bloque será innecesaria solo al final de una secuencia de procesamiento.

Generalmente, este método proporciona una mayor velocidad en comparación con la ausencia de almacenamiento intermedio en el sistema. El proceso de usuario puede procesar un bloque de datos mientras se está leyendo el siguiente. El sistema operativo será capaz de expulsar al proceso porque la operación de entrada tiene lugar dentro de la memoria del sistema en vez de en la memoria de usuario del proceso. Sin embargo, esta técnica complica la lógica del sistema operativo. El sistema operativo debe guardar constancia de las asignaciones de buffers del sistema a los procesos de usuario. La lógica de intercambio también se ve afectada: Si la operación de E/S implica al mismo disco que se usa para intercambio, apenas importa encolar las escrituras al disco necesarios para expulsar al proceso. Este intento de expulsar al proceso y liberar memoria principal no comenzará hasta que la operación de E/S finalice, momento en que la expulsión del proceso al disco puede no ser ya apropiada.

Para la E/S con dispositivos de flujo, el esquema del buffer sencillo puede aplicarse por líneas o por bytes. La operación línea a línea es adecuada para terminales con desplazamiento (scroll) vertical (a veces llamados terminales "tontos"). Con este tipo de terminales, la entrada del usuario se realiza por líneas, marcadas con un retorno de carro al final de la misma. La salida al terminal es similar, línea a línea. Las impresoras de línea constituyen otro ejemplo de tales dispositivos. La operaciones por bytes se utiliza en terminales de pantalla completa, donde cada tecla pulsada tiene su significado, asi como para otros periféricos, como sensores y controladores. En el caso de la E/S por líneas, se puede emplear el buffer para guardar una sola línea. El proceso de usuario quedará suspendido durante la entrada, esperando la llegada de la línea completa. Para la salida, el proceso de usuario puede colocar una línea de salida en el buffer y seguir procesando. No será suspendido a menos que llegue una segunda línea para enviar antes de que se vacíe el buffer de la primera operación de salida. En el caso de la E/S por bytes, la interacción entre el sistema operativo y el proceso de usuario sigue el modelo del productor/consumidor.

Buffer Doble

Se puede realizar una mejora del buffer sencillo asignando dos buffers del sistema a cada operación. De esta forma, un proceso puede transferir datos hacia (o desde) un buffer mientras que el sistema operativo vacía (o rellena) el otro. Esta técnica se conoce como buffer doble o intercambio de buffers.
En la entrada de flujos, se afronta de nuevo el problema de las dos alternativas de operación. Para la E/S de líneas, el proceso de usuario no tiene que ser suspendido para entrada o salida a menos que el proceso se adelante al buffer doble. Para la operación con bytes, el buffer doble no ofrece ninguna ventaja con respecto a un buffer sencillo de doble tamaño. En ambos casos, se seguirá el modelo del productor/consumidor.

Buffer circular

El esquema del buffer doble debería solucionar el flujo de datos entre un dispositivo de E/S y un proceso. Si preocupa el rendimiento de un proceso determinado, sería deseable que las operaciones de E/S fueran capaces de ir al ritmo del proceso. El buffer doble puede ser inapropiado si el proceso lleva a cabo rápidas ráfagas de E/S. En este caso, el problema puede mitigarse usando más de dos buffers.
Cuando se emplean mas de dos, el conjunto de buffers se conoce con el nombre de buffer circular. Cada buffer individual constituye una unidad del buffer circular. Este es, sencillamente, el modelo del productor/consumidor con un buffer limitado.

Spooling

El spooling (Simultaneous Peripheral Operations On-Line) se refiere al proceso mediante el cual la computadora introduce trabajos en un buffer de manera que un dispositivo pueda acceder a ellos cuando esté listo. Se diferencia de los demás métodos principalmente porque este se encuentra en disco y no en memoria principal.

La aplicación más común del spooling es la impresión. En este caso, los documentos son cargados en un buffer, y la impresora los saca de éste a su propia velocidad. El usuario puede entonces realizar otras operaciones en el ordenador mientras la impresión tiene lugar en segundo plano. El spooling permite también que los usuarios coloquen varios trabajos de impresión en una cola de una vez, en lugar de esperar a que cada uno acabe para enviar el siguiente.

Importancia del almacenamiento intermedio

El almacenamiento intermedio es una técnica que soluciona los problemas de “horas punta” en la demanda de E/S. Sin embargo, no existe un tamaño de los buffers que asegure a un dispositivo de E/S ir al mismo ritmo que un proceso cuando la demanda media del proceso es mayor que la que el dispositivo puede admitir. Incluso si se dispone de varios buffers, al final todos se llenaran y el proceso tendrá que quedarse esperando tras operar con una determinada cantidad de datos. Sim embargo, en un entorno de multiprogramación, con la variedad de actividades de E/S y de procesamiento que hay que realizar, el almacenamiento intermedio es una herramienta que puede incrementar la eficiencia del sistema operativo y el rendimiento de los procesos individuales.

Los discos

Son los dispositivos básicos para llevar a cabo almacenamiento masivo y no volátil de datos. Además se usan como plataforma para el sistema de intercambio que usa el gestor de memoria virtual. Son dispositivos electromecánicos (HARD DISK) u

optomecánicos (CD-ROM y DVD), que se acceden a nivel de bloque lógico por el sistema de archivos.

Componentes de un Disco

Desde el punto de vista físico, un disco duro está compuesto por un conjunto de discos que se encuentran en una carcasa cerrada herméticamente junto con una parte mecánica que soporta las cabezas lectoras. Cada uno de esos discos cuenta a su vez con dos caras o superficies sobre las cuales se puede almacenar información. Estas superficies están magnetizadas. El conjunto de cabezales se puede desplazar linealmente desde el exterior hasta el interior de la pila de platos mediante un brazo mecánico que los transporta. Por último, para que los cabezales tengan acceso a la totalidad de los datos, es necesario que la pila de discos gire. Este giro se realiza a velocidad constante y no cesa mientras esté encendido el ordenador.

- Plato: cada uno de los discos de los que están compuesto el disco duro.
- Cara: cada uno de los lados de un plato.
- Cabeza: número de cabezales para la lectura/escritura de información.
- Pista: una circunferencia dentro de una cara.
- Cilindro: conjunto de varias pistas; son todas las circunferencias que están alineadas verticalmente.
- Sector: unidad mínima de información que se puede leer o escribir en un disco duro. Generalmente, cada sector almacena 512 bytes de información

Estructura del Disco Duro

El sistema de almacenamiento secundario se usa para guardar los programas y datos en dispositivos rápidos, de forma que sean fácilmente accesibles a las aplicaciones a través del sistema de archivos. En la jerarquía de E/S se colocan justo debajo de la memoria RAM.
Hay dos elementos involucrados en este sistema:

- Discos. El almacenamiento secundario se lleva a cabo casi exclusivamente sobre discos, por lo que es interesante conocer su estructura y cómo se gestionan.
- Manejadores de disco. Controlan todas las operaciones que se hacen sobre los discos, entre las que son especialmente importantes las cuestiones de planificación de peticiones a disco.

Funciones Principales:

- Proceso de la petición de E/S de bloques.
- Traducción del formato lógico a mandatos del controlador.
- Insertar la petición en la cola del dispositivo, llevando a cabo la política de planificación de disco pertinente (FIFO, SJF, SCAN, CSCAN, EDF, etc.).
- Enviar los mandatos al controlador, programando la DMA.

- Bloqueo en espera de la interrupción de E/S.
- Comprobación del estado de la operación cuando llega la interrupción.
- Gestionar los errores, si existen, y resolverlos si es posible.
- Indicación del estado de terminación al nivel superior del sistema de E/S.

Clasificación de Discos

Según la interfaz de su controlador:

IDE - Integrated Drive Electronics (Electrónica Integrada de Unidad).

- Reúne 25 a 35 sectores en una pista.
- Coloca la unidad y el controlador juntos.
- Se conectan directamente en la placa madre.
- No se le puede dar mantenimiento mediante programas.
- No se puede conectar +2DD.

SCSI – Small Computer System Interface (Interfaz de Sistemas Pequeños de Computo).

- Se puede conectar hasta ocho diferentes tipos (Id0 a Id7).
- Es el único modo de conectar varios DES consistentemente
- Son de > capacidad que los IDE.
- Tienen precios altos y se dificulta su instalación.
- Cada periférico SCSI tiene inteligencia propia, a diferencia de los puertos serie y paralelo.

Según su tecnología de fabricación:

- Discos duros (Winchester).
- Discos ópticos.
- Discos extraíbles.

Independientemente del tipo al que pertenezcan, las estructuras físicas y lógica de todos los discos son muy similares, como se muestra a continuación.

Estructura Lógica de los Discos

• Los DD se manejan como vectores grandes de bloques lógicos, siendo el bloque la unidad mínima de transferencia

• El vector de bloques lógicos se proyecta sobre los sectores del disco secuencialmente:

- Sector 0: primer sector de la primera pista del cilindro más externo.
- El mapa se hace primero en esa pista, luego en las restantes pistas de ese cilindro y luego en los restantes cilindros.
- El manejador de disco no sabe nada de la organización de los ficheros, sólo de particiones y bloques.
- En el sector 0 se guarda la tabla de particiones.
- Particiones activas o de sistema: permiten arrancar desde un sistema operativo.

Planificación de los Movimientos del Brazo del Disco

El tiempo de leer o escribir un sector del disco está determinado por cuatro factores:
m

- Tiempo de latencia: tiempo que se demora en ir de la posición actual hasta una posición adyacente requerida.
- Tiempo de búsqueda: tiempo en mover el brazo desde la posición actual hasta el nuevo cilindro.
- Tiempo de transmisión: tiempo requerido para leer/escribir un registro en disco.

Parámetros de Ejecución de Disco

Para leer o escribir, la cabeza lectora se debe situar en la pista deseada y al comienzo del sector deseado

- Tiempo de búsqueda: Tiempo que toma para colocar la cabeza en la pista deseada.
- Tiempo de latencia (retardo de giro): Tiempo para llegar el inicio del sector a la cabeza lectora.
- Tiempo de acceso = T_búsqueda + T_latencia. Es el tiempo para entrar posición para leer o escribir.
- Tiempo de transferencia: Ocurre cuando el sector deseado se mueve debajo de la cabeza. T_acceso = (n * T_búsqueda) + T_latencia + T_transferencia.

Cache de disco

Una cache de disco es un buffer para los sectores de disco situados en la memoria principal. La
cache contiene una copia de algunos sectores del disco. Cuando se hace una petición de E/S para un sector específico, se comprueba si el sector está en la caché del disco. Si es así, la petición se cumple con la caché. Si no, se lee el sector pedido del disco y se coloca en la caché. Debido a la cercanía de referencias, cuando se traiga un bloque de datos a la caché para sastifacer una sola petición de E/S, será probable que se produzcan referencias futuras al mismo bloque.

Son de interés varias cuestiones de diseño. En primer lugar, cuando una petición de E/S se sirve con la caché, los datos de la misma deben ser enviados al proceso que los solicito. El Envio puede hacerse por una transferencia en memoria del bloque de datos, desde la caché del disco a la memoria asignada al proceso de usuario o, simplemente, usar la posibilidad de memoria compartida y pasar un puntero a la entrada apropiada de la caché del disco. Este último método ahorra el tiempo de la transferencia interna en memoria y, además, permite el acceso compartido de otros procesos.

Estrategia de reemplazo.

Cuando se trae un nuevo sector a la caché del disco, uno de los bloques existentes debe ser sustituido.
El algoritmo más común es el de "el usado hace más tiempo" (LRU)

En el que el bloque que ha permanecido sin referencias en la caché por más tiempo es reemplazado. Lógicamente, la caché constará de una pila de bloques, estando situado el más reciente en la cima de la pila. Cuando se referencia un bloque de la caché, se le mueve de su posición a la cima de la pila. Cuando se trae un bloque de la memoria secundaria, se elimina el bloque que está en el fondo de la pila, colocando al recién llegado en la cima de la pila. Naturalmente, no es necesario mover estos bloques por la memoria: puede asociarse una pila de punteros a la caché.

Algoritmo de "la menos usada" (LFU).

Se sustituye el bloque de la caché que ha sufrido un menor número de referencias. El algoritmo LFU podría implementarse asociando un contador a cada bloque. Cuando se trae un bloque, se le asigna un valor 1: con cada referencia al bloque, se incrementa el contador en una unidad. Cuando hace falta un reemplazo, se selecciona el bloque con menor valor del contador. Intuitivamente, prodia parecer que el LFU es mas adecuado que el LRU porque se emplea mas información de cada bloque en el proceso de selección.

El sencillo algoritmo de LFU tiene el siguiente problema. Puede ser que ciertos bloque se referencien poco frecuentemente, pero cuando lo son, se produzcan intervalos cortos de referencias repetidas, debido a la cercanía, obteniéndose así grandes valores del contador de referencias. Tras este intervalo, el valor del contador de referencias puede ser engañoso y no reflejar la probabilidad de que el bloque sea referenciado nuevamente. De este modo, el efecto de la cercanía puede originar que el algoritmo LFU realice malas elecciones en el reemplazo

Para superar esta dificultad del LFU, se propone una técnica conocida como *reemplazo en función de la frecuencia.*

Los bloques están organizados lógicamente en una pila, como en el algoritmo LRU una parte determinada de la cima de la pila es reservada como una *sección nueva.* Cuando se hace blanco en la caché, el bloque referenciado es trasladado a la cima de la pila. Si el bloque ya está en la sección nueva, su contador de referencia no se incrementa suficientemente grande, el resultado de este procedimiento es que el contador de los bloques referenciados repetidamente en un corto intervalo de tiempo permanece inalterado. Si se produce una falta, se elegirá para reemplazar el bloque con el menor valor del contador de referencias que no esté en la sección nueva; en caso de empate, se elegirá el usado hace más tiempo

Esta estrategia sólo se consiguió una leve mejora sobre el LRU. El problema consiste en lo siguiente:

1. Cuando se produzca una falta de caché, se traerá un nuevo bloque a la sección nueva, con un contador de uno a uno.
2. El contador permanece a uno mientras el bloque se quede en la sección nueva.
3. El bloque envejece y, finalmente, sale de la sección nueva con su contador todavía a uno.
4. Si el blooque no vuelve a ser referenciado rápidamente, es muy probable que sea reemplazado porque, forzosamente, posee el menor contador de todos los bloques que no están en la sección nueva. En otras palabras, no parece haber un intervalo suficientemente grande para que los bloques que salen de la seccion nueva aumenten sus contadores, incluso si han sido referenciados más o menos frecuentemente

La sustitución puede llevarse a cabo bajo demanda o planificada previamente.
Por demanda: los sectores se sustituyen sólo cuando se necesita su entrada de la tabla.

Planificada previamente: cada vez se liberan un conjunto de entradas. La razón de ser de este método está relacionada con la necesidad de volver a escribir los sectores. Si se trae un sector a la caché y sólo es leído, no será necesario volver a escribirlo al disco cuando sea reemplazado. Sin embargo, si el sector es actualizado, entonces sí será necesario volver a escribirlo antes de reemplazarlo. En este último caso, tiene sentido agrupar las escrituras y ordenarlas para minimizar el tiempo de búsqueda.

Planificación de Disco

.

El tiempo de acceso tiene dos componentes principales:

- búsqueda: tiempo que tarda el brazo del disco para mover las cabezas hasta el cilindro que contiene el sector deseado.
- latencia: tiempo de espera adicional para que el disco gire hasta ponerse sobre el sector deseado.
- Objetivo: minimizar el tiempo de búsqueda, que es directamente proporcional a la distancia de búsqueda. Es claro que una política de planificación debe intentar maximizar la capacidad de ejecución, el número de peticiones servidas por unidad de tiempo. Debido a la planificación se reduce el tiempo desperdiciado en las esperas de las búsquedas, con lo que se puede mejorar la media de los tiempos de respuesta.

•Ancho de banda: bytes Ancho de banda transferidos/tiempo de transferencia.

Algoritmos de Planificación de disco

Planificación FCFS

La forma más sencilla de planificación de disco es, desde luego, el servicio por orden de llegada (**FCFS**, first come, first served). No proporciona el servicio más rápido.
La planificación FCFS es justa en el sentido de que una vez que llega una petición, se fija su lugar dentro de la cola de espera. Una petición, se fija su lugar dentro de la cola de espera. Una petición no puede ser desplazada por la llegada de otra con prioridad más alta.
La FCFS es aceptable cuando la carga en un disco es ligera. Pero a medida que crece la carga, la FCFS tiende a saturar el dispositivo y los tiempos de respuesta se incrementan. La FCFS ofrece una varianza pequeña, pero tiene tiempos de espera muy grandes.

Planificación SSTF (Shortest-Seek-Time First)

Parece razonable atender todas las solicitudes cercanas a la posición actual de la cabeza antes de mover la cabeza a una posición lejana para atender otras solicitudes. Este supuesto es la base del algoritmo de tiempo de búsqueda más corto primero (**SSTF**, shortest-seek-time-first), que selecciona la solicitud que tiene el menor tiempo de búsqueda a partir de la posición actual de la cabeza.
En esta política la petición que da por resultado la distancia de búsqueda más corta (y, con esto, el tiempo de búsqueda más corto) es la siguiente en ser servida, aunque esa petición no sea la primera en la cola.
Los patrones de búsqueda SSTF tienden a estar muy relocalizados, dando como resultado que las pistas internas y externas reciban un servicio pobre, en comparación con las pistas del centro. La SSTF es útil en sistemas de procesamiento por lotes, en los cuales la capacidad de ejecución es lo más importante. Pero la alta varianza de los tiempos de respuesta (es decir, su falta de predecibilidad) lo hace inaceptable para los sistemas interactivos.
La planificación SSTF es en esencia una forma de planificación de trabajo más corto primero (SJF) y, al igual que la planificación SFJ, puede cause inanición de algunas solicitudes.
Aunque el algoritmo SSTF representa una mejora sustancial respecto al algoritmo FCFS, no es óptimo.

Planificación SCAN

En el algoritmo SCAN, el brazo del disco parte de un extremo del disco y se mueve hacia el otro, atendiendo las solicitudes a medida que llega a cada cilindro, hasta llegar al otro extremo del disco. Ahí, la dirección de movimiento de la cabeza se

invierte, y continúa la atención. La cabeza barre continuamente el disco de un lado a otro.
Esta política, desarrollada por Denning, opera como SSTF, excepto que selecciona la petición que da como resultado la distancia de búsqueda más corto en una dirección seleccionada. La SCAN no cambia de dirección hasta que ha alcanzado el cilindro exterior o hasta que ya NO haya peticiones pendientes en la dirección con preferencia.
La SCAN se comporta de manera parecida al SSTF desde el punto de vista de la mejora en la capacidad de ejecución y de la media de los tiempos de respuesta, pero elimina mucha de la discriminación inherente a los esquemas SSTF y ofrece una varianza menor.
El algoritmo SCAN también se conoce como algoritmo de elevador, ya que el brazo del disco se comporta igual que el elevador de un edificio, que atiende primero todas las solicitudes para subir y luego cambia de dirección para atender las solicitudes de abajo.

Planificación C-SCAN

La planificación SCAN circular (C-SCAN) es una variante de SCAN diseñada para dar un tiempo de espera más uniforme. Al igual que SCAN, C-SCAN mueve la cabeza de un extremo del disco al otro, atendiendo las solicitudes en el camino, sólo que ahora, cuando la cabeza llega al otro extremo, regresa de inmediato al principio del disco sin atender solicitudes.
El algoritmo de planificación C-SCAN básicamente trata los cilindros como una lista circular que continúa del último cilindro al primero.
En la estrategia C-SCAN, el brazo se mueve del cilindro exterior al interior, sirviendo a las peticiones con menor tiempo de búsqueda. Cuando el brazo ha completado su recorrido hacia adentro, salta a la petición más cercana al cilindro exterior y a continuación reanuda su recorrido hacia adentro procesando peticiones.
La C-SCAN puede implementarse de forma que las peticiones que llegan durante un recorrido sean servidos en el siguiente. De esta forma C-SCAN elimina completamente la discriminación contra las peticiones para los cilindros exterior e interior. Tiene una varianza de los tiempos de respuesta muy pequeña.

Planificación LOOK y C-LOOK

Se diferencian con la planificación SCAN y C-SCAN porque lo más común es que el brazo solo vaya hasta el cilindro correspondiente a la solicitud final en cada dirección, entonces allí invierte su dirección inmediatamente sin llegar al extremo del disco. Estos algoritmos miran (look) si hay una solicitud antes de continuar moviéndose en una determinada dirección.

Consideraciones de Rendimiento

La E/S es uno de los factores que más afectan al rendimiento del sistema. El gran problema de todos estos dispositivos de E/S es que son muy lentos. Mientras la CPU procesa instrucciones a casi 1 GHz y la memoria RAM tiene un tiempo de

acceso de nanosegundos, los dispositivos de E/S más rápidos tienen una velocidad de acceso del orden de milisegundos. Esta diferencia en la velocidad de acceso, y el hecho de que las aplicaciones son cada vez más interactivas y necesitan más E/S, hace que los sistemas de E/S sean el cuello de botella más importante de los sistemas de computación y que todos los sistemas operativos dediquen un gran esfuerzo a desarrollar y optimizar todos los mecanismos de E/S. Piense, por ejemplo, que el mero hecho de seguir el curso de un ratón supone inspeccionar su posición varias veces por segundo. Igualmente, los dispositivos de comunicaciones interrumpen continuamente el flujo de ejecución de la CPU para comunicar la llegada de paquetes de datos.

Jerarquía de dispositivos de E/S según su velocidad de acceso

Evaluación del rendimiento

El acceso y transferencia de un determinado bloque del disco conlleva una sucesión de operaciones con retardos asociados, algunos de ellos considerables al ser de naturaleza mecánica. Los tiempos que componen una operación de acceso a un bloque son los siguientes:

(1) Tiempo de posicionamiento sobre el cilindro, tpos. Depende del recorrido que tenga que hacer el brazo de cabezales (distancia al cilindro desde la posición actual). Para la estimación del tiempo de acceso vamos a suponer un tiempo medio (que denotaremos Tpos), dependiente de la estrategia de planificación de accesos que se siga.

(2) Selección de pista. Se trata de seleccionar uno de los cabezales del brazo.

Como únicamente supone la activación de una señal eléctrica, este retardo no se considera.
(3) Tiempo de posicionamiento en el comienzo del primer sector del bloque. Está relacionado con la velocidad de giro y varía desde 0 al tiempo de un giro completo, tgiro. En media hay que considerar este retardo como de 0,5tgiro.

(4) Tiempo de posicionamiento en el resto de los sectores (para bloques de más de un sector). Los accesos a los demás sectores están condicionados por la posición del cabezal tras la transferencia del sector anterior, lo que depende del factor de

interleaving o, alternativamente, de que el tamaño del buffer permita almacenar todos los sectores del bloque. En un disco de S sectores, sin buffers y con factor de interleaving F, el tiempo de posicionamiento en cada uno de los N–1 últimos sectores de un bloque de N sectores será Ftgiro/S.

(5) Tiempo de acceso a cada sector. Depende de la velocidad de giro y del número de sectores por pista. Para cada sector del bloque es tgiro/S.

(6) Tiempo de transferencia de sector, ttrans. Involucra una operación de DMA y depende de múltiples factores, como la velocidad del bus, la existencia o no de buffers en el controlador o el funcionamiento del DMA. Este tiempo se solapa con el posicionamiento en el siguiente sector, de forma que sólo hay que considerar el tiempo de transferencia de un sector por bloque, independientemente de su tamaño.

Además, habría que considerar la incidencia de los retardos debidos al tratamiento de errores (por ejemplo, puede ser necesario posicionar más de una vez el cabezal), que depende de las probabilidades de cada tipo de error. Simplificando, para bloques de N sectores, con factor de interleaving F, sin considerar el tratamiento de errores, ni la existencia de buffers en el controlador, ni solapamiento entre operaciones, se obtiene la siguiente expresión del tiempo medio de acceso al bloque:

$$Tacc = Tpos + (0,5 + (N–1)F/S + N/S)\ tgiro + ttrans$$

Esta expresión responde a un modelo simplificado con las condiciones comentadas. En otros casos, la expresión del tiempo de acceso debe deducirse de acuerdo a los parámetros definidos para el sistema particular.

Se pueden emplear diversos principios para mejorar la eficiencia de la E/S:

- Reducir el número de cambios de contexto.
- Reducir el número de veces que los datos deben copiarse en memoria mientras pasan desde el dispositivo a la aplicación o viceversa.
- Reducir la frecuencia de las interrupciones utilizando transferencias de gran tamaño, controladoras inteligentes y mecanismos de sondeo (si puede minimizarse la espera activa).
- Incrementar la concurrencia utilizando controladoras preparadas para DMA o canales, con el fin de descargar a la CPU de las operaciones simples de copia de datos.
- Desplazar las primitivas de procesamiento al hardware, para permitir que se ejecuten en las controladoras de dispositivo, de forma concurrente con las operaciones de la CPU y del bus.
- Equilibrar el rendimiento de la CPU, del subsistema de memoria, del bus y de la E/S, porque cualquier sobrecarga en una de esas áreas provocará la aparición de tiempos muertos en las otras.

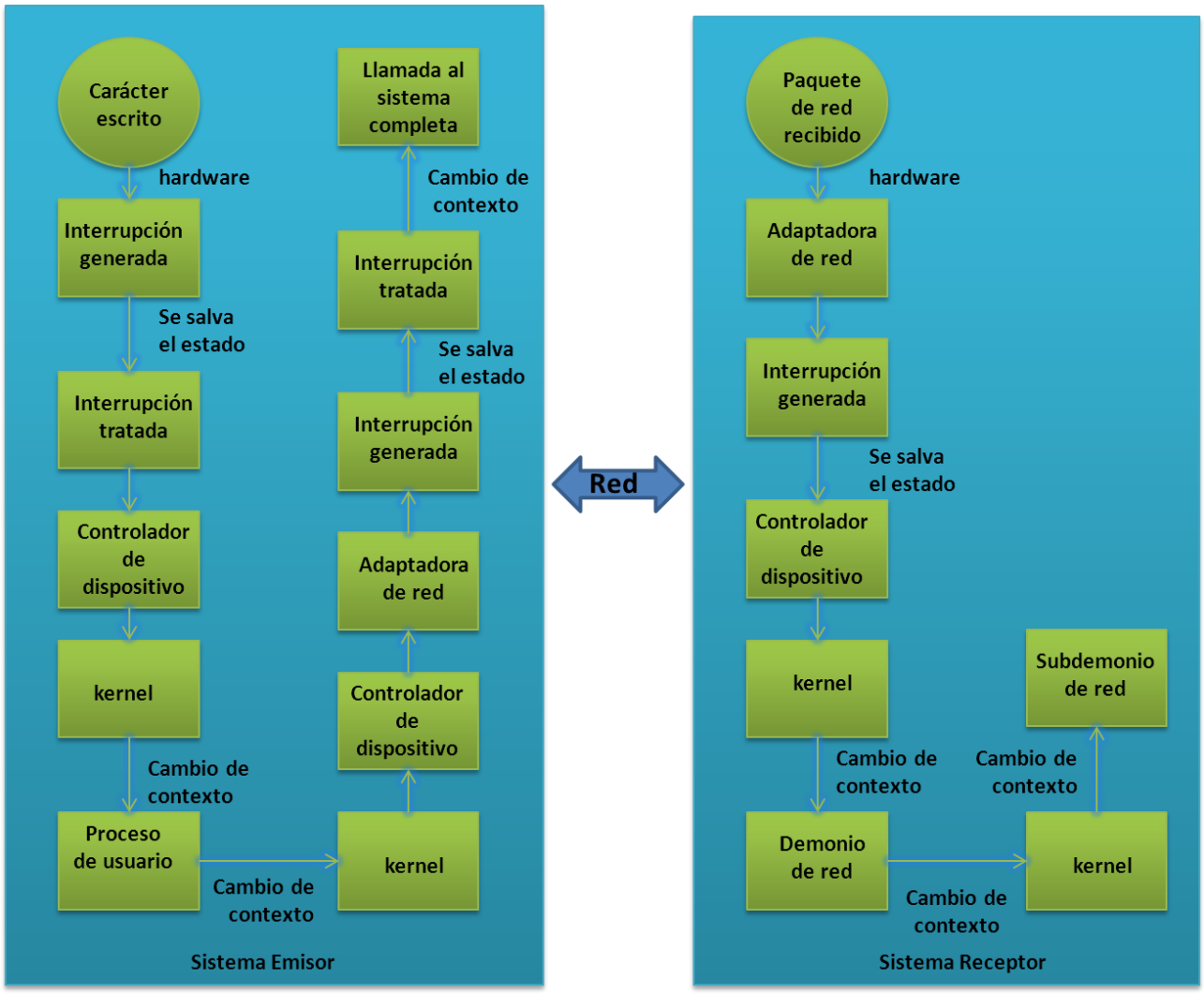
El tráfico de red también puede provocar una alta tasa de cambios de contexto. Considere, por ejemplo, un inicio de sesión remoto en una máquina desde otra. Cada carácter escrito en la máquina local debe transportarse hasta la máquina remota. En la máquina local, el carácter se escribe en el teclado, se genera una interrupción de teclado y el carácter se pasa a través de la rutina de tratamiento de interrupción al controlador de dispositivo, al *kernel y* luego al proceso de usuario. El proceso de usuario ejecuta una llamada de E/S de red al sistema para enviar el carácter a la página remota. El carácter fluye entonces hacia el *kernel* local, a través de los niveles de red que construyen un paquete de red y hacia el controlador del dispositivo de red. El controlador del dispositivo de red transfiere el paquete a la tarjeta controladora de red, que envía el carácter y genera una interrupción. La interrupción pasa de nuevo a través del *kernel para,* hacer que se complete la llamada de E/S de red al sistema como se observa en la figura de abajo.

Ahora, el hardware de red del sistema remoto recibe el paquete y se genera una interrupción. El carácter se desempaqueta según los protocolos de red y se entrega al demonio de red apropiado. El demonio de red identifica qué sesión remota es la implicada y pasa el paquete al subdemonio apropiado para dicha sesión. A lo largo de este flujo, se producen cambios de contexto y cambios de estado**).**

Usualmente, el receptor devuelve el carácter como eco al transmisor, y dicha operación duplica la cantidad de trabajo que hay que realizar.

Para eliminar los cambios de contextos necesarios para mover cada carácter entre los demonios del *kernel,* los desarrolladores de Solaris reimplementaron el demonio telnet utilizando hebras internas al *kernel.* Sun estima que esta mejora permite incrementar el número máximo de inicios de sesión de red desde unos cuantos centenares a unos cuantos miles en un servidor de gran tamaño.

Otros sistemas utilizan procesadores frontales separados para la E/S de terminales con el fin de reducir la carga de interrupciones en la CPU principal. Por ejemplo, un concentrador de terminales puede multiplexar el tráfico de centenares de terminales remotos con un único puerto en una computadora de gran tamaño.



Referencias bibliográficas

- William Stallings, (1997) Sistemas Operativos, (2ed). Madrid: Pearson Educación, S.A.

- Silbershatz, Galvin, Gagne (2005). Fundamentos de Sistemas Operativos (Séptima Edición). México: Editorial McGraw-Hill.

- Jesús Carretero, Pedro De Miguel, Félix García, Fernando Pérez: “Sistemas Operativos Una visión aplicada”, Editorial Mg Graw Hill.