UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

A C A T L A N

" SISTEMAS OPERATIVOS "

T E S I N A

QUE PARA OBTENER EL TITULO DE

LIC. MATEMATICAS APLICADAS Y COMPUTACION

P R E S E N T A

MARITZA NOVA JUAREZ

ACATLAN, EDO. DE MEXICO 1993

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE

Prologo

Este trabajo fué desarrollado con el objeto de brindar un mayor apoyo a los profesores, alumnos y todo el personal involucrado en el proceso de preparación de los estudiantes de la Licenciatura de Matemáticas Aplicadas y Computación. Pues es responsabilidad de todos que el nivel de los egresados de esta carrera sea uno de los mejores a nivel nacional, dado que las condiciones actuales requieren de un conocimiento más amplio en el campo de la computación, pues en esta época es aplicable prácticamente en todas las áreas del conocimiento.

Durante el tiempo que he pertenecido a la UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO he participado en dos papeles, tanto a nivel docencia y como alumno, lo que me permitió tener una visión más amplia de los planes de estudio que se deben de seguir para una mayor competividad en el campo laboral.

El contenido que pongo a consideración de todos intenta sugerir a los profesores las alternativas que en mi experencia profesional me han sido de utilidad para una actualización constante que beneficie tanto a profesores como a alumnos en la adquisición de conocimientos que ésten al día, principalmente de aquellos que sean parte importante en el desarrollo de un profesionista dentro del mercado de trabajo, lo que redunda en un mejor desarrollo personal dando un mayor prestigio a nuestra escuela.

Tambien es mi intencion que este trabajo sirva de base para enriquecer los planes de estudio de nuestra carrera con los propósitos que anteriormente se mencionan.

Introducción

A lo largo de mi desarrollo escolar y laboral he ido descubriendo la importancia y trascendencia de conocimientos sólidos y actualizados de todas las materias impartidas en la carrera de Matemáticas Aplicadas y Computación para el buen desempeño de las actividades profesionales de sus egresados, pero principalmente de las materias relacionadas con la computación ya que para los estudiantes y egresados de MAC es vital no sólo el manejo de las computadoras sino también la optimización de los recursos que estas nos proporcionan, ya que aproximadamente un 90% de los egresados de la carrera sin importar su preespecialidad, Laboran en el área de Sistemas Computacionales. Los Sistemas Operativos se encargan de distribuir el uso de los recursos de las computadoras, y evolucionan a medida que el usuario requiere de una mayor optimización en sus sistemas de información (Manejar una cantidad mayor de información en un menor tiempo de procesamiento).

Debido precisamente a esto el presente trabajo tiene como objetivos:

1) Proponer el programa para la materia de Sistemas Operativos que este de acuerdo con las necesidades que impone el mercado de trabajo del egresado de MAC.

2) Proporcionar un material de apoyo bibliográfico para el estudio de la materia considerando las características de formación del estudiante de la carrera.

Para cumplir con sus objetivos se recomienda tener conocimientos previos de Datos y estructuras Almacenamiento ya que éstos permitirán al lector una comprensión más rápida de los Sistemas Operativos.

El trabajo consta de 4 capítulos:

1.- Arquitectura del Sistema Operativo. Incluye una breve perspectiva histórica, el concepto, las funciones, características de diferentes tipos de Sistemas Operativos. Y se presenta además el modelo de estudio sobre el cual se basa el trabajo (estructura del S.O).

2.- Estructura de un Sistema Operativo. En este capítulo se desarrollan los temas del modelo de estudio propuesto.

3.- Estudio del Sistema Operativo UNIX. Se presenta un análisis de los componentes de dicho sistema desarrollandolo según la estructura del capítulo 2, el cual se complementa con una lista de sus principales comandos y funciones.

4.- Propuesta de programa para la impartición de la materia de Sistemas Operativos. Se hacen algunas observaciones sobre el actual temario de la materia, y se realiza una propuesta de temario además de hacer recomendaciones para la impartición de la misma.

CAPITULO 1.

Arquitectura del Sistema Operativo

El objetivo de este capitulo es presentar un marco de referencia para el estudio de los Sistemas operativos Multiusuarios, incluye una breve perspectiva, la definición, las funciones, las caracteristicas, así como la definición de los diferentes tipos de sistemas operativos que existen. Se presenta además el modelo de estudio sobre el cual se basa el trabajo.

1.1 DEFINICION Y FUNCIONES DE UN SISTEMA OPERATIVO.

Para introducirnos al estudio de los Sistemas Operativos es necesario tener una idea clara de lo que este término implica. Por lo cual a continuación se presentan definiciones de varios autores. Un Sistema Operativo es:

1)" Conjunto de programas que controlan la operación automática de un sistema de cómputo, con el fin de optimizar el funcionamiento y presentar una imagen monolitica y virtual ante sus usuarios." (1)

2)" La reunión de dos conjuntos de programas: los de control y los de proceso."(2)

3)" Un conjunto de programas que manejan o supervisan el funcionamiento o la operación de la computadora."(3)

4)" Coleccion de programas y utilerias diseñadas para facilitar al usuario la creación, el manejo de archivos, de los programas y la operación de los dispositivos del sistema conectados a la computadora."(4)

5)" Interfaz entre la computadora y el usuario que proporciona al usuario un control flexible y cómodo sobre los recursos de la computadora."(5)

Como se observara todas las definiciones anteriores hacen referencia a la palabra control, además mencionan que el sistema operativo sirve para realizar una interfaz con el operador. Por lo que podemos englobar la definición de un Sistema Operativo en la siguiente formula:

SISTEMA OPERATIVO = PROG. CONTROL + PROG. DE PROCESO =>
... INTERFAZ DEL USUARIO

De lo cual se puede concluir que el Sistema Operativo es responsable de la operación eficiente de la computadora. De la misma forma en que a un empleado se le pueden dar Instrucciones acerca de los trabajos a efectuarse primero, o bien el momento de interrumpir un trabajo para realizar otro mas importante, un sistema de computo necesita un conjunto de instrucciones que dirijan sus actividades. De aqui la importancia de un sistema operativo y la razon por la que se dice que éste es parte esencial de un sistema de computo.

1 Levine Guillermo introducción a la computación
2 Salas Parrilla Jesus Sistemas Operativos y compiladores
3 D Irwin Richard Principles of data processing
4 Duffy Tim Four Software tools Plus
5 Ben-Ari, M. Principles of Concurrent Programming

Funciones

Los sistemas operativos actuales generalmente presentan las siguientes funciones :

- Asignación de tareas al procesador o los procesadores.
- Asignar localidades de memoria y otras áreas de almacenamiento.
- Manejar las transiciones de tarea a tarea.
- Interpretar órdenes o instrucciones.
- Coordinar el uso de compiladores, ensambladores, programas de utilería y otros dispositivos de programas.
- Establecer un sistema de prioridades para el desempeño de tareas.
- Implementar forzosamente el sistema de prioridades de tareas y mantener la disciplina y el orden en todo sistema de cómputo.
- Desempeñarse como cronómetro de tiempo interno.
- Transferir el control del sistema de cómputo a los programas de computadora en la secuencia y el tiempo apropiados.
- Definir itinerarios para las tareas y trabajos de procesamiento.
- Coordinar y asignar los dispositivos de entrada y salida, en lo que se están ejecutando uno o más programas.
- Producir vaciados, rastreos, mensajes en error y otras asistencias para corregir y detectar errores.
- Establecer integridad y seguridad de los datos.
- Evitar la interferencia de unos programas con otros.
- Comunicarse con el operador de la computadora.
- Coordinar la transferencia de archivos y datos de un dispositivo de almacenamiento.

Las anteriores funciones pueden resumirse en una función central :

ADMINISTRAR Y ORGANIZAR LOS RECURSOS DE QUE DISPONE UNA COMPUTADORA PARA SU UTILIZACION BRINDANDO SERVICIO A UN MAYOR NUMERO DE USUARIOS.

1.2 CARACTERISTICAS

Las compañías que elaboran sistemas operativos consideran que las exigencias que debe cumplir todo sistema operativo son:

- Debe permitir la concurrencia de procesos cubriendo el tiempo de E/S y ocupando la memoria con varios programas.

- Tener la posibilidad de compartir recursos, a distintos niveles.

- Que cualquier proceso pueda ejecutarse en el momento que se solicite, si existen recursos libres para él.

- Eficiencia en el sistema: Esta se calcula considerando los siguientes puntos:

  1. Tiempo medio que ocupa cada trabajo.

2. Tiempo que el CPU no es empleado.
3. Tiempo de respuesta en sistemas de multiacceso.
4. Tiempo de utilización de recursos en tales como CPU y dispositivos de E/S.
5. Tiempo de plazo *entre* dos asignaciones de CPU a un mismo programa.

- Un sistema operativo tiene que ser fiable es decir no debe tener errores y debe preveer todas las situaciones.

1.3 PERSPECTIVA HISTORICA.

En los comienzos de la historia de las computadoras destacaron Howard H. Aiken, quien en 1944 construyó la primera computadora electromecánica MARK-I en la Universidad de Harvard. John W. Manchly y J. Presper Eckert Jr. construyeron en la Universidad de Pennsylvania la primera computadora electrónica a base de válvulas de vacío ENIAC .Estas computadoras no tenían sistema operativo ya que todas sus operaciones se cableaban, tenían un manejo de switches, sus usuarios desarrollaban en lenguaje de máquina.

Primera Generación (década de los cincuentas)

Se crea el primer sistema operativo S.O.S. (Shared Operating System) creado por General Motors para una IBM 701. Diseñado para hacer más *fluida* la *transición entre* trabajos. Permitía el arranque y pagar la computadora, realizar trabajos automáticamente, limpieza de memoria automática, y manejo básico de recursos.

Antes de que los sistemas operativos fueran diseñados, se perdía un tiempo entre la terminación de un trabajo y el inició del siguiente. Este fué el comienzo de los sistemas de procesamiento por lotes, donde los trabajos se *remiten* por *grupos* o lotes. Cuando el trabajo estaba en ejecución éste tenía el control de la máquina. Al terminar cada trabajo el control era devuelto al sistema operativo, el cual limpiaba, leía e iniciaba el trabajo siguiente.

Segunda Generación (1960 - 1965)

a) La característica de esta generación de sistemas operativos fue el desarrollo de los sistemas compartidos con multiprogramación, y los principios del multiprocesamiento. En los sistemas de multiprogramación, varios programas de usuarios se encuentran al mismo tiempo en el almacenamiento principal, y el procesador se cambia *rápidamente* de un trabajo a otro. En los sistemas de multiprocesamiento se utilizan varios procesadores en un solo sistema computacional, con la finalidad de incrementar el poder de procesamiento de la máquina. En los sistemas operativos de la *primera* generación cuando el usuario deseaba escribir datos en una cinta tenía que hacer referencia específica a una unidad de cinta particular, en los sistemas de segunda generación, el programa del usuario especificaba tan sólo que un archivo iba a ser *escrito* en una unidad de cinta con cierto número de pistas y cierta densidad. El sistema operativo localizaba entonces, una unidad de cinta disponible con las características deseadas, y le indicaba al operador que montara una cinta en esa unidad.

b) Se desarrollaron sistemas de tiempo compartido, en los que los usuarios podían acoplarse directamente con el computador a través de terminales parecidas a máquinas de escribir. Los sistemas de tiempo compartido operan un *modo interactivo* o conversacional con los usuarios. El usuario tecleaba una petición al computador, éste la procesaba tan pronto como le era posible, y la respuesta (si la había) se desplegaba en la terminal del usuario. La computación conversacional hizo posible grandes adelantos en el proceso de desarrollo de programas. Un usuario de tiempo compartido podía, en segundos o minutos, localizar y corregir errores; mientras que, en un ambiente de procesamiento por *lotes*, el usuario tenía que sufrir retrasos, a menudo de horas o *hasta* días.

c) Surgieron sistemas de tiempo real, que fueron utilizados en el control de procesos industriales, como en la refinación de gasolina. Los sistemas militares de tiempo real fueron desarrollados para regular o supervisar, en caso de sufrir un ataque aéreo miles de puntos al mismo tiempo.

Tercera Generación (1965 - 1975)

Esta generacion inicia con la introducción de la familia de computadoras Sistema/360 de IBM. Las computadoras de la tercera generación fueron diseñadas como sistemas de uso generales. Los sistemas operativos de la tercera generación eran sistemas de modos múltiples. Algunos soportaban simultáneamente procesos por lotes, tiempo compartido, procesamientos de tiempo real y multiprocesamiento. Estos sistemas introdujeron mayor complejidad a los ambientes computacionales; a la cual en un principio, no estaban acostumbrados los usuarios. Los sistemas interponían una capa de software entre el usuario y el hardware. Esta capa de software a veces era tan gruesa que el usuario perdía de vista al hardware, y veía sólo el punto creado por el software. Para lograr que uno de estos sistemas realizara la tarea más simple, los usuarios debían familiarizarse con un complicado lenguaje de control de trabajos, a fin de poder especificar el trabajo y los recursos requeridos. En resumen en la tercera generación :

- Los sistemas trabajan en multimodo (trabajar al mismo tiempo en Batch, Multiproceso y multireal).
- Avance en telecomunicaciones .
- Manejo de bases de datos (concurrencia para una misma base de datos).
- Computación Distribuida (no se encarga del proceso completamente sino que lo parte y varias computadoras se encargan de él).
- Procesos en tiempo real (varias computadoras manejan varios procesos juntos dando respuesta inmediata).
- Los sistemas operativos son bastante costosos en cuanto a su desarrollo y venta.
- Inicia la estandarización en los Sistemas Operativos.

Cuarta Generación (1975 a nuestros días).

Los sistemas operativos de la cuarta generación constituyen el estado actual de la tecnología computacional. Con la ampliación del uso de redes de cómputo y del procesamiento en línea los usuarios obtienen acceso a computadoras alejadas geográficamente a través de varios tipos de terminales. El microprocesador ha hecho posible la aparición de la computadora personal con las cuales muchos usuarios han desarrollado sistemas de computación que son accesibles para su uso personal en cualquier momento del día o de la noche. Las computadoras personales están equipadas con interfaz para comunicación de datos , y también sirven como terminales. El usuario de un sistema de cuarta generación ya no sólo se comunica con una computadora en un modo de tiempo compartido, en lugar de esto, el usuario puede comunicarse con sistemas alejados geográficamente.

Tendencias futuras en los Sistemas Operativos:

1. El hardware de las computadoras continuara declinando su precio, mientras que las velocidades de procesamiento y capacidades de almacenamiento aumentan, y tamaño físico de los procesadores y memorias disminuye.

2. Los sistemas basados en multiprocesamiento se hacen cada día más comunes.

3. Los lenguajes de programación están siendo desarrollados con el fin de explotar la concurrencia, el hardware y los sistemas operativos están siendo diseñados para ejecutar con mayor eficiencia los programas concurrentes.

4. El procesamiento masivo en paralelo se volverá común. Será imposible ejecutar programas a mayor velocidad, debido al alto grado de concurrencia.

5. Las computadoras y los sistemas operativos estarán diseñados para fomentar la operación de máquinas virtuales. Las computadoras estarán conectadas cada vez más en redes de sistemas lo cual continuara enfatizando la importancia de las perspectivas de las máquinas virtuales.

6. El concepto de proceso distribuido provocará que sean desarrollados sistemas operativos dispersos, en los cuales las funciones de los sistemas operativos son distribuidas entre varios procesadores a través de grandes redes de sistemas.

7. La perspectiva previa de un sistema operativo como administrador de recursos perdurara en el futuro, pero los recursos administrados, en especial los datos, serán considerados cada vez más como un recurso para ser administrado.

8. El manejo de Sistemas Operativos cada vez incrementa más el uso de interfases gráficas.

9. Los problemas de seguridad se incrementarán debido a que la información pasa a través de varios tipos vulnerables de líneas de comunicación.

1.4 TIPOS DE SISTEMAS OPERATIVOS.

A) SISTEMAS EN BATCH:

Es un sistema que ejecuta bloques de procesos secuencialmente y no comienza con uno nuevo hasta haber terminado el anterior.

Los primeros sistemas Batch se realizaron entre 1956-1958 para poder tener:

- Mejor uso del equipo.
- Mayor velocidad del proceso.
- Menor costo.

En estos sistemas no existe la multiprogramación.

La relación : Usuario - Job - Job Steps - Proceso es 1 a 1 es decir es un proceso en línea.

Ventajas:

- Simplificación de implementación.
- Más fácil manejo de recursos.
- Facilidad en su uso.
- No hay multiprogramación.

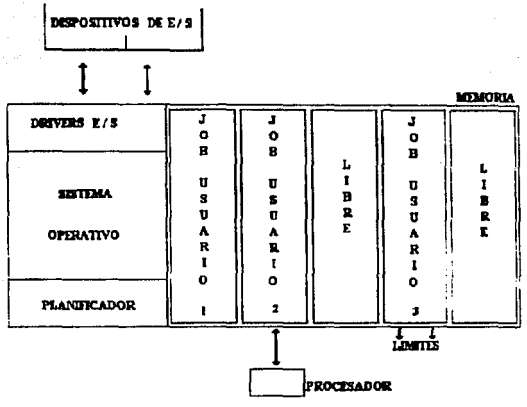
Desventajas:

- Velocidad.
- Uniproceso.
- Subutilización de recursos para flexibilidad.

B) SISTEMAS DE MULTIPROGRAMACION:

En estos sistemas se pueden ejecutar varios programas simultáneamente con el fin de aprovechar al máximo los recursos de la computadora. La multiprogramación surgió de la imposibilidad para los sistemas o modos de trabajo en los que con un solo trabajo se podía tener ocupados al procesador y a los dispositivos de entrada/salida durante todo el tiempo.

Sistemas de multiprogramación (Fig 1.1)



Un trabajo realizado en una computadora, desde el punto de vista de ocupación en tiempo del procesador y los dispositivos periféricos, puede ser de dos tipos:

a) Trabajos limitados por proceso: Son aquellos que consumen la mayor parte de su tiempo en el tratamiento de la información y muy poco en operaciones de entrada/salida.

b) Trabajos limitados por operaciones de entrada/salida: Son los que dedican la mayor parte de su tiempo en operaciones de entrada/salida, haciendo poco uso del procesador, que se mantiene inactivo durante grandes periodos de tiempo. Existen dos problemas dentro de este tipo de sistemas operativos:

- Relocalización: Cómo llevar control del punto en memoria donde se encuentra el proceso.

- Protección: Cómo restringir el acceso a memoria de un proceso (Mantenerlo dentro de ciertos límites).

1) Sistemas de Tiempo Real:

Son una modalidad de los sistemas operativos multiprogramados, en los cuales se necesita un tiempo de respuesta pequeño ante cualquier peticion. Suele emplearse en aplicaciones dedicadas a sistemas de control con sensores como elementos de entrada, donde es necesaria una respuesta rápida sobre el sistema a controlar. Un sistema trabaja en tiempo real si el tiempo de respuesta permite controlar y regular al medio sobre el que opera. Las caracteristicas principales de tiempo real son:

- Fuertes restricciones en el tiempo de respuesta (milisegundos).
- La información debe estar permanentemente actualizada.
- El sistema debe permanecer practicamente inactivo para atender lo más *rápidamente* posible cualquier evento en la entrada.
- Manejo eficaz de interrupciones.
- Manejo sencillo de prioridades.
- Gestión de memoria real.

2) Sistemas de Tiempo Compartido:

En este modo de trabajo la organización no se realiza por trabajos sino por sesiones. Una sesión es todo un conjunto de trabajos que se realizan desde que un usuario se conecta a la computadora hasta que se despide de la misma. Durante estas sesiones se pueden realizar multitud de operaciones controladas por un proceso denominado intérprete de comandos, que mantiene el diálogo entre el usuario y el sistema operativo. Este proceso puede dar lugar a otros muchos para realizar todas las demandas del usuario. Los sistemas de tiempo compartido se caracterizan por:

- Ser muy conversacionales.
- Atender a varios usuarios simultáneamente.
- Ofrecer tiempos de respuesta relativamente cortos (segundos).
- Mantener una interrogación secuencial de peticiones de usuarios (polling).
- Poseer una fuerte gestión de archivos.
- Utilizar técnicas de buffering y spooling.
- Gestionar memoria virtual.

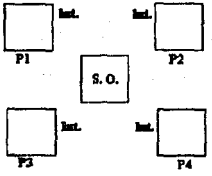
3) Sistemas de Multiprocesamiento:

Los Sistemas de Multiprocesamiento trabajan en paralelo, y son independientes unos de otros. Los avances en Sistemas Operativos de *Multiproceso* han sido muy lentos principalmente debido a:

- La mente humana no está preparada para el multiproceso.
- Falta de lenguajes para *multiproceso* tales como Ada, Pascal concurrente o Modula.
- Hay poca experiencia en paralelismo.
- El hardware es secuencial.
- La depuración de programas en ejecución.
- Es difícil comprobar la correcta ejecución de un programa.

Existen tres tipos de sistemas operativos básicos en multiproceso:

a) Maestro - Esclavo.



En este tipo el procesador maestro ejecuta el sistema operativo, los procesadores esclavos ejecutan los trabajos de los usuarios y el procesador maestro no se puede caer.

Ventajas:

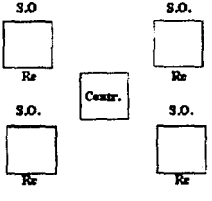
- Las tablas de sistema operativo son de más fácil manejo.
- Tiene procesadores redundantes.
- Hay una carga específica de trabajo.

Desventaja:

- El procesador maestro debe atender a otros procesadores (baja velocidad de proceso).

b) Ejecutivos Separados.

Hay n procesadores, cada uno con la copia del sistema operativo y atendiendo a cada proceso.



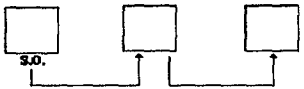
El controlador asigna recursos específicos a cada uno.

Ventajas:
- Es un sistema más confiable.
- La información global del sistema está al alcance de cualquier procesador.

Desventaja:

- Posibilidad de subutilización del equipo.

c) **Simétrico.**



En estos sistemas la información se encuentra de procesador en procesador y sólo se tiene una copia del *sistema operativo*. Hay una serie de dispositivos y cada proceso corre un mismo proceso que se puede partir en microinstrucciones.

Ventajas:

- Rapidez.
- Disminución en la carga de ejecución.

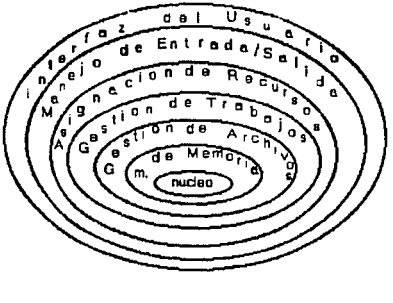
Desventajas:

- Bloqueo de un proceso (hasta que se termine el *anterior* proceso el siguiente proceso no puede ejecutarse), los procesadores deben esperar a que el otro proceso termine con el bloque de instrucciones que se está ejecutando.

1.5 MODELO DE ESTUDIO (ESTRUCTURA).

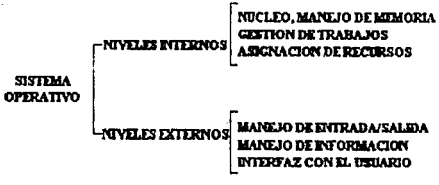
Una vez definido el concepto de un sistema operativo se presenta un esquema del mismo:

(figura 1.1)



El esquema propuesto se basa en la estructura de cualquier sistema operativo multiusuario. Consta básicamente de dos niveles:

(figura 1.2)



El nivel más interno de un sistema operativo es el nucleo, ya que es el más cercano al hardware; éste se constituye por un conjunto de programas que se encargan del manejo del procesador. También son niveles internos:

- El manejo o administración de memoria; se encarga de manejar los contenidos de memoria del sistema.
- La gestión de archivos se compone de una serie de rutinas de tratamiento de archivos externos.

- La gestión de trabajos y la asignación de recursos que definen tareas para el sistema a partir de los datos de entrada.

A partir del nivel de manejo de E/S se dice que los niveles son externos o virtuales ya que se encuentran más cercanos al usuario, éstos regulan los niveles internos, lo que significa que los externos, darán toda la información que necesiten los internos.
Los niveles externos son:

- El control de las entradas y salidas del sistema, la corrección de las mismas y otras funciones para el manejo de datos.
- El manejo de información permite a los usuarios el manejo libre de cualquier cantidad de información que se desee almacenar, leer, imprimir o desechar.
- La interfaz con el usuario es el último nivel externo y básicamente es un lenguaje de control que el usuario entiende o un conjunto de menús que el sistema despliega en la pantalla, para que el usuario seleccione la operación que desee efectuar.

CAPITULO 2.

Estructura de un Sistema Operativo (Modelo de Estudio)

En el presente capítulo se realiza un estudio teórico de los componentes de cualquier sistema operativo multiusuario, el cual se basa en el modelo propuesto en el capítulo anterior.

Para poder iniciar el estudio de los componentes de un Sistema Operativo se introducirán conceptos elementales relacionados al mismo.

Recurso: Es la actividad o capacidad de un componente físico de la computadora como la memoria, *CPU* o los dispositivos periféricos.

Proceso o tarea: Es una secuencia temporal de ejecuciones de instrucciones.

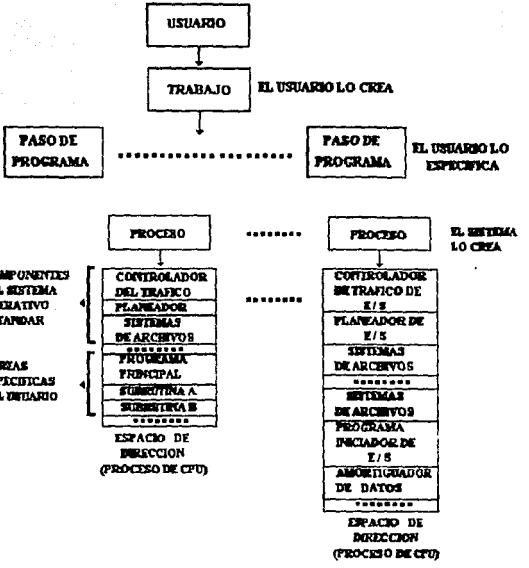
Trabajo: Es un componente de tareas o procesos que pertenecen al mismo usuario.

Usuario: Cualquiera que realice una actividad empleando una computadora.

Espacio de dirección: Conjunto de programas y datos que son accesados en un proceso.

La siguiente figura ilustra la relación entre los componentes descritos anteriormente:

Relacion entre usuario, trabajo proceso y espacio de direccion
(figura 2.1)



El Sistema Operativo hace que un proceso pase por una serie de transiciones antes de terminar su tarea. Dichas transiciones se denominan Estados de transición de un proceso:

El usuario entrega su trabajo al sistema que consiste en varios lotes de programas precedidos por instrucciones de control de trabajo, estas instrucciones solicitan los recursos que necesita el sistema operativo.

La rutina spool lee el trabajo y lo coloca en un disco. (Para encontrar el espacio de almacenaje llama a la administración de información que lleva la cuenta de toda la información y espacio disponible en el sistema).

Despues la rutina de planeación de trabajos busca todos los archivos en el disco y selecciona un trabajo para ser admitido por el sistema. Al elegir el trabajo, el planeador llama primero al administrador de memoria para determinar si existe suficiente memoria principal y después al administrador de los dispositivos para determinar si los que necesita están disponibles.

Una vez que el planificador de trabajos determina los recursos que necesita, se llama al controlador de tráfico para que cree la información asociada al proceso y a la administración de memoria para asignar la memoria principal necesaria.

Ya cargado el proceso en memoria éste se encuentra listo para correrse. Si el proceso de corrida solicita la lectura de un archivo, la administración de información llama la administración de los dispositivos para iniciar la lectura del archivo (operación de entrada/salida) y luego llama a el administrador de procesos para indicar que el proceso que solicita el archivo está aguardando que se complete el tiempo de espera.

Cuando éste concluye el hardware de entrada/salida envía una señal al controlador de tráfico en la administración de procesos y lo prepara para correr otra vez.

Si el proceso finaliza al correr nuevamente, entonces se le coloca en el estado terminal y se liberan todos los recursos asignados.

Esquemas de estados de transición (figura 2.2)

COMPONENTES DE UN SISTEMA OPERATIVO

2.1 NUCLEO.

El nucleo es la parte mas interna del conjunto de programas que forman un sistema operativo, su función principal es tomar el control del procesador y determinar cuándo y cómo lo va a repartir entre los usuarios. Muchas veces el núcleo es conocido también como el administrador de procesos.

Las finciones del núcleo son:

- Llevar control del registro de los procesadores y el estado de los procesos al controlador de tráfico.
- Decidir quien utilizará el procesador.
- Asignar el recurso (procesador) a un proceso asignando los registros necesarios de hardware .
- Recuperar el recurso (procesador) cuando el proceso cede el uso del procesador, aborta o excede la cantidad permisible de utilización.

La administración del procesador es importante cuando los recursos de la computadora se comparten. Este requerimiento proviene de dos fuentes: Multiprogramación y tiempo compartido.

La multiprogramación es un sistema de explotación que intenta mejorar el rendimiento de los diferentes componentes de un equipo de cómputo. Trata de evitar el tiempo en que el CPU no trabaja en espera de operaciones de entrada y salida. La multiprogramación permite la ejecución de dos o más programas simultáneamente entendiendo por esto pasar el control de ejecución de un programa a otro cuando el primero entra en estado de espera. En otras palabras la multiprogramación es la ejecución concurrente de dos o más programas. (Un procesamiento concurrente es cuando dos o más programas están activos dentro del mismo marco de tiempo pero no utilizan el mismo recurso de la computadora en el mismo instante).

En un ambiente de multiprogramación, los programas parecen ejecutarse simultáneamente desde el principio hasta el final, pero realmente están siendo interrumpidos constantemente a medida que la computadora cambia de un programa a otro.

En el sistema de tiempos compartidos todos los programas que se encuentran en la memoria se consideran como si estuvieran ejecutándose. Cada programa se explota, segim su turno asignado por la unidad central durante un tiempo prefijado; al cabo del mismo se produce una interrupción forzosa y se pasa a ejecutar el programa siguiente.

El tiempo compartido es similar a la multiprogramación por el hecho de que los recursos se comparten, pero en este caso es en base a un evento. En un ambiente de tiempo compartido, el sistema operativo controla a los usuarios asignándoles tiempo del procesador.

El núcleo tiene un doble papel; simular una actividad concurrente para los procesos y proporcionar la interfaz con el hardware. El núcleo convierte los programas de los usuarios en procesos para el sistema. Este toma los programas origioales y les asigna una representación interna que permite que el sistema operativo determine los recursos que los procesos requieran de la computadora. Todo esto con la finalidad de que el sistema operativo pueda realizar una planeación eficiente de la distribución de los recursos de cómputo entre los diversos usuarios. En términos generales sucede que el procesador abandona el proceso que esta siendo ejecutado, y dedica su atención a ejecutar otro, cuando el primero entra en algún estado de espera.

El núcleo se compone de:

- **Despachador** (dispatcher) es una rutina de planificación de las actividades del CPU. Cada vez que uno de los trabajos no puede o no debe continuar, el CPU recibe el control para elegir el proceso que debe ejecutarse a continuación.

- **Manejador de interrupciones** (supervisor) que dirige la acción del sistema al producirse una interrupción que señala el fin de un proceso o hecho anormal.

- **Rutinas de apertura y cierre** (Controlador de tráfico) que se ejecutan cuando varios procesos quieren utilizar al mismo tiempo el mismo recurso. Este es utilizado solamente por un proceso; así que el resto de los procesos espera a que quede libre dicho recurso.

A continuación se explica el funcionamiento de cada uno de ellos.

2.1.1 DESPACHADOR (Planeador de nivel bajo):

El despachador (dispatcher o scheduler) determina que proceso asignar al procesador que ha quedado libre y si se puede o no asignar. Las funciones principales del despachador son:

- Llevar el control del estado de los procesos.
- Asignar procesadores a los procesos.
- Desasignar los procesadores de los procesos.

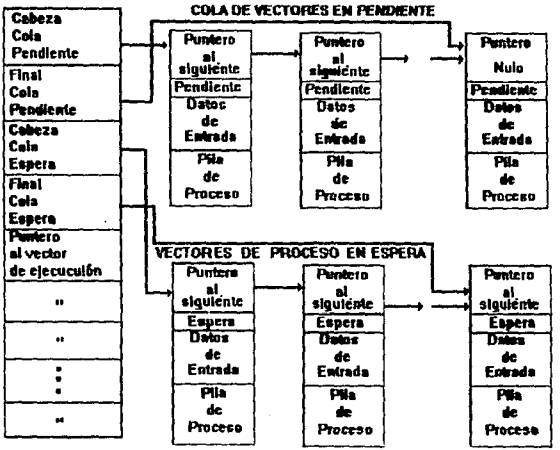
Para cumplir con sus funciones el despachador maneja una cola de procesos que está formada por una tabla y un grupo de vectores de estado de los procesos. Esta cola entra en ejecución cuando se produce un cambio de estado en algún proceso. En estas circunstancias el despachador debe corroborar si el proceso puede o no seguir. De no seguir, va a la cola de procesos, donde checa a cuál de ellos asignara el procesador que ha quedado libre.

La cola de procesos tiene una entrada horizontal por procesador y en ella se indica el proceso que esta ejecutando dicho procesador. El grupo de vectores de estado es el que básicamente forma las colas de procesos. Una de éstas está formada por todos los vectores que pertenecen a los procesos que están en el estado de pendiente. Otra cola de procesos está formada por los vectores de los procesos que están en espera de realizar una operación de entrada/salida. Así pues el despachador tiene dos tipos de colas: La de pendientes y la de espera.

Por ejemplo cuando se está ejecutando algo en el procesador 1 y pasa del estado de ejecución al de espera. Provoca que el despachador guarde el estado del procesador 1 en la pila del proceso que estaba ejecutando y asigne al procesador 1 el primer proceso de la cola de pendiente, con lo que el segundo proceso pasa a ser el primer estado del nuevo proceso del procesador 1 (en estado de ejecución) el estado del viejo proceso del procesador 1 cambia al de espera y ocupa la última posición de la cola de espera.

Muchas veces se dice que el despachador es el corazon del sistema ya que este checa en forma secuencial las terminales para determinar si requieren servicio. Si se encuentra una terminal con una solicitud pendiente, el sistema lee el mensaje proveniente de dicha terminal, carga el módulo que se ocupa de ese tipo de solicitud y lo inicia. El módulo puede regresar al despachador si requiere conducir un intercambio con la terminal, o puede llamar a otro módulo para que continue el procesamiento.

Estructura de datos que maneja un despachador (figura 2.3)



2.1.2 MANEJO DE INTERRUPCIONES

Su función principal es manejar las interrupciones del procesador central. En un sistema operativo se tiene un conjunto de operaciones que puede efectuar el procesador, éstas se conocen como privilegiadas. Una operación privilegiada es aquella que al ser ejecutada causa que el procesador entre en un estado de interrupción (por ejemplo una operación de entrada/salida). Durante este estado el procesador se detiene momentáneamente y se cuestiona si debe o no seguir ejecutando la operación. En otras palabras el procesador ejecuta automáticamente un programa de atención a la interrupción que investiga la causa de ésta y determina los pasos a seguir dependiendo de la naturaleza de la misma.

Existen cuatro posibilidades:

1. Interrupción anticipada: es el resultado de una acción no esperada por el procesador.

2. La interrupción, que puede o no ser anticipada, incluye sólo mantenimiento. La rutina de interrupción efectúa el mantenimiento y después continúa con el proceso interrumpido.

3. Cuando la interrupcion requiere actividad sustancial, por ejemplo, si se solicita una transferencia de disco, se crea un proceso y se agrega a la lista del controlador de tráfico.

4. La interrupción surge a partir de una contingencia. En este caso se llama un procedimiento adecuado del núcleo que crearía un nuevo proceso de ser necesario.

Cuando un proceso pide la ejecución de una operación privilegiada, inmediatamente el procesador realiza una interrupción e instantáneamente ocurre un desvío en la secuencia de ejecución de instrucciones, y en vez de continuar ejecutando el programa, el procesador ejecuta una rutina que atiende la interrupción. Esta rutina es un programa del sistema que averigua la causa del desvío, determina para ese caso particular cual es la solución e indica al procesador lo que debe hacer a continuación. Lo cual depende de si hay o no otros procesos participando en la multiplexión de la unidad central de procesamiento. Si no existe otro proceso en espera del procesador, el control vuelve al proceso original, y el procesador espera a que se complete la operación deseada. Si existen más procesos en estado de espera, entonces el proceso original se congela, en tanto que otro proceso se descongela y recibe el control sobre el procesador.

Dentro del núcleo se tienen 4 tipos de interrupciones:

1) Programas.
2) Interrupciones de error de máquina.
3) Interrupciones externas.
4) Interrupciones de llamada al supervisor.

1) PROGRAMAS.

Las interrupciones de programas son aquellas que pueden producirse durante la compilación o corrida de un programa, dentro de éstas existen varias clases:

a) Aritméticas.
- Desbordamiento (overflow): Resultado sobre el rango definido en la aritmética de punto flotante.
- División por cero.
- Vacío (Underflow): Resultado por debajo del rango definido en la aritmética en punto flotante.

b) Intento de acceso a memoria protegida.
- Este tipo de interrupción provoca siempre un error insalvable y el sistema operativo da por terminado el proceso.

c) Anomalía de direccionamiento en la memoria.

2) INTERRUPCIONES DE ERROR DE MAQUINA

Las interrupciones de error de máquina se provocan siempre por un error en cualquier componente del hardware del sistema ya sea debido al operador o a un error de programa, entre ellas se encuentran los siguientes ejemplos:

- Error de paridad de la memoria central.
- Fallo de alimentación.
- Fallo en la transmisión entre canales y memoria central.

3) INTERRUPCIONES EXTERNAS.

Entre las interrupciones externas se encuentran:
- Reloj de intervalos.
- Botón de interrupción del procesador (reset).
- Interrupción de comunicaciones de CPU a CPU.
- Operaciones de entrada/salida, tales como comandos inválidos de E/S, terminación de canal de E/S y terminación de dispositivo de E/S.

Para el control de las interrupciones externas el sistema operativo tiene un supervisor que esta dividido en dos grupos de programas:

- Programas de preservación de registro.
- Programas de diagnóstico de interrupciones.

El sistema de interrupciones externas de cualquier CPU consta de un conjunto de líneas de interrupción y otras tantas direcciones de memoria asociadas. Cuando en una de estas líneas se presenta una interrupción, el procesador finaliza la instrucción en fase de ejecución, almacena en memoria su estado en ese instante y bifurca a la posición de memoria asociada a la línea. El programa obtenido a partir de esta dirección se llama rutina de servicio de la interrupción. Se realiza la *ejecución* de la rutina que al concluirse restaura el estado que el CPU tenía en el momento de la interrupción y por consiguiente, reanuda el *programa interrumpido*.

4) INTERRUPCIONES DE LLAMADAS AL SUPERVISOR.

Las interrupciones de llamada al supervisor son aquellas que el programador produce en su programa, es decir el programa hace uso del sistema de interrupciones en beneficio de su ejecución. Cuando el computador tiene varias líneas de interrupción, existe un orden de prioridad de servicio fijo, asignado por el *hardware*. De manera que se resuelven los conflictos que surgen cuando hay señales simultáneas de interrupción en más de una línea. Primero se atiende a la de mayor prioridad (si se está atendiendo una línea de interrupción y se presenta otra línea con mayor prioridad se interrumpe la primera y se atiende la segunda al terminar ésta se reanuda la anterior).

2.1.3 RUTINAS DE APERTURA Y CIERRE (SINCRONIZACION).

Su función es controlar el tráfico del procesador esté es, coordinar los diversos procesos del sistema operativo y de los usuarios que interactúan en el núcleo del sistema para que el CPU no se confunda. Una forma de llevar el control del estado de un proceso es utilizar una base de datos asociada con cada proceso en el sistema denominado bloque de control de proceso (PBC): este bloque existe en cada proceso. Con frecuencia se enlazan entre sí todos los *bloques* de control de procesos en el mismo estado, y a la lista resultante se le conoce como lista de bloques o preparados. El controlador de tráfico es *llamado siempre* que cambia el estado de un recurso. Es posible subdividir más la lista de bloques si se da una cadena para cada causa por la que se bloquée un proceso. Por consecuencia si un proceso solicita un dispositivo que ya esté ocupado, se enlaza aquel a la cadena de bloques asociada con ese dispositivo. Cuando se libera el dispositivo, el controlador de tráfico comprueba su cadena de bloques para determinar si hay procesos en espera del mismo. En caso afirmativo, se vuelven los procesos al estado listo para volver a solicitar el dispositivo. En otros casos, se puede asignar el dispositivo a uno de los procesos y colocarlo en estado listo, mientras que los demás procesos que esperan al dispositivo están bloqueados. Una rutina de apertura o cierre debe decidir el proceso que se debe asignar a un procesador y durante cuánto tiempo.

La duración de asignación de un procesador a un proceso puede depender de una o alguna combinación de lo siguiente:

- El proceso ha terminado.
- El proceso está bloqueado.
- Un proceso de mayor prioridad necesita el procesador.
- Ha transcurrido un intervalo de tiempo.
- Ocurre un error.

Entre los problemas a los que se enfrentan la *rutinas de apertura* y cierre se encuentra el de sincronización de procesos que se deriva de la necesidad de compartir recursos en un sistema de cómputo. Por sincronización se entiende la *garantía* del orden cuando éste se conoce previamente. Una buena sincronización debe impedir cualquier tipo de conflicto en un conjunto de procesos concurrentes, esto es, debe impedir el comienzo de ejecución de un proceso hasta que se tengan las condiciones adecuadas para que tal ejecución pueda realizarse y no llegue a producir conflictos. En la sincronización se puede presentar una *condición de carrera* cuando la planeación de dos procesos es tan crítica que las distintas órdenes de plancarlos producen distintos cómputos. En estas condiciones se deben compartir datos o recursos en forma explícita o implícita entre dos o más procesos. Suponganse dos procesos que ocasionalmente solicitan la impresión de una línea en una sola impresora.

Dependiendo de la planeación de los procesos 1 y 2 todo el proceso de impresión 1 puede adelantarse o seguir a la impresión del proceso 2. Pero es muy probable que la impresión de cada proceso esté intercalada con el otro en el listado. Una solución a este problema es requerir que un proceso solicite explícitamente usar el recurso compartido antes de utilizarlo. Al terminar todo el uso del recurso, el proceso puede liberarlo.

Se dispone de varios mecanismos de sincronización para proporcionar la coordinación y la comunicación entre procesos, entre éstos se encuentran los siguientes:

a) Instrucción de prueba y fija.

En ésta se *utiliza* una entidad física para representar al recurso, que a menudo se conoce como byte de candado o semáforo. Un semáforo puede ser un simple conmutador "semáforo binario" o puede tener un valor entero "semáforo contador". Un semáforo esté asignado con cada recurso en el sistema. Cuando un recurso es requerido, el operador P es usado para examinar su correspondiente semáforo; si éste está *apagado*, se enciende y regresa; si se encuentra encendido entonces P notifica que el proceso debe ser colocado en un estado de espera asociado con el semáforo, después de que un operador V es enviado por otro proceso liberando el recurso, apagando el semáforo y notificando a procesos que estén en espera. Por ejemplo si tenemos la siguiente condición: si el recurso estádisponible el semáforo=0 y cuando el recurso se encuentre ocupado el semáforo=1. Antes de operar el recurso compartido, un proceso debe realizar las siguientes acciones:

1. Examinar el valor del semáforo o byte del candado (es 0 ó 1).
2. Prender el semáforo en 1.
3. Si el valor original era semáforo=1, regresar al paso 1.

Después que el proceso termina de usar el recurso, apaga el semáforo dándole un valor de 0. Este mecanismo es suficiente para sincronización pero su desventaja es el desperdicio de los recursos del procesador, esto se debe a que el proceso bloqueado realmente no se detiene, sino que éste cicla continuamente probando el semáforo y esperando que cambie a cero.

b) *Mecanismos de espera y señal.*

Este mecanismo surge como una mejora al de instrucción de prueba fija, para evitar un desperdicio de recursos del procesador, definiendo mecanismos modificados de bloqueo y desbloqueo. Sea X el semáforo; se llama BLOQUEO(X) a la acción del uso de un recurso compartido y a la acción después del uso es DESBLOQUE(X).

BLOQUEO(X)

1. Examinar el valor del semáforo (0 ó 1).
2. Prender el semáforo a 1.
3. Si el valor original es 1, llamar ESPERA(X).

DESBLOQUE(X)

1. Apagar el semáforo (semáforo = 0).
2. Llamar SEÑAL(X).

ESPERA(X) asocia el bloque de control de procesos a un estado bloqueado y lo enlaza con el semáforo X. Entonces se selecciona otro proceso que debe correr el planeador de procesos.

SEÑAL(X) comprueba la lista bloqueada asociada con el semáforo X; de haber procesos bloqueados en espera de X, se selecciona uno y se prende su bloque de control de procesos al estado de listo. Se pueden usar los mecanismos ESPERA y SEÑAL para otros propósitos, tales como para esperar la terminación de una solicitud de Entrada/Salida.

c) Operaciones de P y V en semáforos de conteo.

Los mecanismos BLOQUEO y DESBLOQUEO son conocidos como operaciones P y V. Estas operan en semáforos de conteo, que son variables que toman valores enteros, no solamente 0 y 1. Los mecanismos se pueden definir como:

P(S)

1. Disminuye el valor de S (por ejemplo S=S-1).
2. Si S es inferior a 0, ESPERA(S).

V(S)

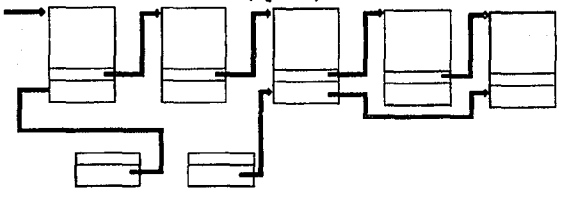
1. Incrementa el valor de S (por ejemplo S=S+1).
2. Si S es menor, o igual a 0, SEÑAL(S).

Dependiendo del valor inicial de los semáforos y de la cantidad de semáforos empleados, se pueden utilizar P y V para varios propósitos. Si se emplea un semáforo y su valor inicial es 1, P(S) y V(S) son iguales a ESPERA(S) y SEÑAL(S).

En concreto, un semáforo es una variable especial con propiedades ya descritas. En la práctica es necesario saber cuándo un proceso está esperando un semáforo, de forma que pueda ser liberado cuando otro proceso envíe una señal a ese semáforo. Una solución a esto consiste en agregar otro campo al descriptor de procesos para que contenga la cadena de alto y encadenar unos a otros todos aquellos procesos que esperen un semaforo especifico. El semáforo se convierte en un elemento de dos componentes que contiene el valor entero y la cabeza de la cadena de alto. Estructuras de datos para el implante practico de semáforos

Estructuras de datos para el implante práctico de semáforo

(figura 2.4)



Una limitación de los semáforos es que un proceso puede detenerse sólo por una razón cada vez: no puede pedir que se le detenga hasta que ocurra un cierto número de eventos. Este problema tiene una posible solución que consiste en encadenar unos a otros los descriptores de procesos que tengan una necesidad de tiempo de paro e incluir en el descriptor un conteo denominado reloj local.

2.2 MANEJO DE MEMORIA.

El objetivo de esta parte es enfatizar el papel de la administración de memoria con relación al procesamiento del trabajo. El empleo óptimo de la memoria de la computadora es un problema importante en los sistemas que emplean la multiprogramación ya que en ocasiones se requiere más memoria que la disponible. Debido a la diversidad de requerimientos de un sistema de cómputo existen varias técnicas específicas de manejo de memoria, algunas de ellas son:

- Administración de memoria contigua simple.
- Administración de memoria por particiones.
- Administración de memoria por particiones relocalizables.
- Administración de memoria paginada.
- Administración de memoria paginada por demanda.
- Administración de memoria segmentada.
- Administración de memoria segmentada y paginada.

La administración de memoria de un sistema operativo se refiere a la memoria primaria que accesa directamente los procesadores para tener sus instrucciones y datos.

Las funciones principales del administrador de memoria son:

1) Tener un registro exacto del recurso (memoria); siempre lleva la cuenta del estado de cada posición de la memoria primaria.

2) Decidir que proceso tendrá el control de la memoria; en otras palabras determina la política de asignación para la memoria. Si un proceso está compartiendo memoria primaria con uno o más procesos entonces la administración de recursos debe determinar los requerimientos para cada proceso.

3) Asignar el recurso: Esto es, una vez decidida la asignación de memoria el administrador debe seleccionar las posiciones específicas y actualizar la información de asignación.

4) Manejar la recuperación de la memoria (técnica y política de desasignación): Un proceso puede liberar la memoria asignada antes o la administración de memoria puede reclamar la memoria en base a alguna política de desasignación.

En resumen la función principal del administrador de memoria es la asignación de los programas a localidades de la memoria, el movimiento de segmentos de programas y datos hacia y desde la memoria interna y los dispositivos de almacenamiento externo.

Requerimientos para el manejo de memoria.

- **Protección**: Aún si sólo una imagen de proceso se encuentra en la memoria en un momento dado, comparte ésta con el núcleo, y es imprescindible que una falla del proceso nunca pueda escribir sobre el código del núcleo o de datos. Luego entonces si más de una imagen de proceso se encuentra en memoria en un momento dado, existe la necesidad de hacer forzosa la protección mutua entre ellos.

- **Transparencia**: la asignación de memoria debe ser *invisible* para el proceso. Si un proceso se detiene durante un tiempo prolongado, tal vez se desee sacarlo del disco. La asignación de memoria debe ser tal que los procesos puedan correrse hacia una área diferente de memoria sin ninguna acción especial dentro del proceso.

- **Segmentos múltiples**: la imagen del proceso está compuesta lógicamente por un número de segmentos: código, datos, pila y datos del sistema. Puede requerirse tener estos segmentos en área de memoria *físicamente* separadas aun cuando el código, la pila, y los segmentos de datos deban aparecer como lógicamente contiguos.

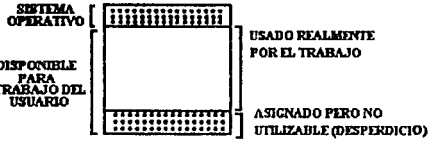
- **Código compartido**: si el código del programa es invariante, esto es, si no se ve alterado por el programa, y si tal programa es simultáneamente parte de dos o más imágenes de proceso puede resultar conveniente conservar una sola copia física del segmento de código, que aparece lógicamente en todas las *imágenes* adecuadas del proceso.

2.2.1 Administración de memoria contigua simple

La asignación contigua simple no requiere de características especiales de hardware; *usualmente* se asocia con minicomputadores y con pequeños sistemas operativos en lotes. A veces es deseable tener un *mecanismo* de protección de hardware de tipo primitivo para asegurar que los programas del usuario no interfieren en forma accidental con el sistema operativo, que puede consistir en un registro de *límites* y un modo de supervisor-usuario del CPU. El registro de límites contiene la dirección del área protegida. Si el CPU está en modo usuario en cada referencia a la memoria, el hardware hace comprobaciones para asegurarse que no se trate de un acceso al área protegida. De intentarse un acceso, ocurre una interrupción y el control se transfiere al sistema operativo.

En estos sistemas no hay multiprogramación aunque si existe una correspondencia biunívoca entre un usuario, un trabajo y un proceso. Se pueden usar en forma intercambiada los términos usuario, trabajo y proceso.

Asignación contigua simple (figura 2.5)



La memoria está dividida conceptualmente en tres campos contiguos. Una porción de la memoria está disponible al único trabajo que se esté procesando. El trabajo físicamente sólo usa una porción de la memoria asignada dejando una región asignada y no utilizada de memoria.

Características:

- La memoria se asigna por completo a un trabajo.
- Cuando el trabajo termina, toda la memoria se libera.

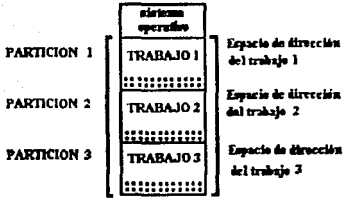
La ventaja principal de este método radica en su simplicidad. Un sistema operativo que lo emplea apenas requiere de 1k bytes, y no se necesita de mucha habilidad para comprenderlo o usarlo. Su desventaja principal es que la memoria no se utiliza al máximo. Esto es debido a las siguientes ineficencias:

- La memoria no se usa del todo. (ver fig 2.5)
- En la mayoría de los sistemas con canales cuando el CPU no está usando toda la memoria, por ejemplo si el trabajo ha iniciado una solicitud de E/S y espera a que el canal complete sus operaciones. La memoria que contiene los programas en el CPU del usuario no se está utilizando mientras el CPU espera.
- Ofrece poca flexibilidad ya que el tamaño total del programa debe ser más pequeño que la memoria principal, pues de lo contrario no se pueden correr los trabajos.
- Existen porciones de memoria que jamás se accesan; por ejemplo las rutinas de error que crean los programas al momento de correrse.

2.2.2 Administración de memoria por particiones

El manejo de memoria por particiones consiste en subdividir la memoria en varias secciones fijas y asignar cada una de ellas a un usuario o proceso activo. Su principal función es asegurar que ningún usuario intervenga en el área de memoria asignada a otro.

Asignación de memoria particionada (figura 2.6)



En este tipo de gestión de memoria el tiempo de CPU se asigna a cada programa según su esquema de división de tiempos. La CPU pasa a otro programa, si el que se estaba ejecutando cumple estas condiones:

- El programa se sustituye o elimina de la memoria.
- Se termina el programa si el error es irrecuperable.
- Petición de entrada/salida.
- El tiempo de CPU asignado a ese programa es agotado.

Dadas las condiciones anteriores se puede concluir que esta gestión de memoria trabaja en tiempo compartido.

Características:

- Mantiene un registro del estado de cada partición.
- El planeador de trabajos determina principalmente quién obtiene la memoria.
- Para cada trabajo asigna una partición disponible de tamaño suficiente.
- Al terminar el trabajo, se indica que la partición no está en uso y que está disponible para una asignación futura.

Este método requiere de muy poco hardware especial. Es deseable tener un mecanismo de protección de memoria para impedir que un trabajo afecte al sistema operativo o a otros trabajos, sea de manera intencional o no. Para evitar esto se pueden usar dos registros de límites para encerrar la partición que se esté utilizando, así si el trabajo tratara de accesar la memoria fuera de la partición, ocurrirá una interrupción de protección aunque esto presenta dos desventajas:

a) Los registros de límites deben cambiarse cada vez que se reasigna el procesador para la multiprogramación.
b) Es difícil extender la protección al canal de E/S. Por ejemplo un trabajo en la partición 1 podría intentar dar instrucciones al canal para que leyera datos en un área de la partición 3. Se podrían proporcionar registros de frontera para cada canal de E/S, requiriendo registros múltiples para canales multipléxor. En forma alterna el sistema operativo que inicia todas las operaciones de entrada y salida, podría tratar de verificar el programa del canal mediante un programa, lo que sería un trabajo muy lento. Las ventajas fundamentales de este modelo son:

- Permite la multiprogramación y facilita la utilización más eficiente del procesador.

- No requiere de hardware especial y costoso.
- Los algoritmos empleados son simples y fáciles de implementar.

Una de sus desventajas es que deja lugares libres en la memoria que, como son de tamaño fijo, no pueden ser utilizados más que por procesos de longitud menor o igual a la de la partición referida. Al seleccionar un algoritmo de memoria particionada se debe tener en cuenta la velocidad y simplicidad del mismo. Esto debido a la fragmentación que es su problema principal y ésta es debida al desarrollo de un número grande de áreas libres por separado, es decir la memoria libre total se fragmenta para construir pedazos pequeños. Aunque es posible superar la fragmentación usando técnicas distintas a la asignación particionada; es posible minimizar este problema mediante una cuidadosa selección de los algoritmos específicos a usar. Esta técnica de administración de memoria tiene muchas variantes pero las más comunes son :

1) Especificación de partición estática.

Esta especificación implica que la memoria se divida en particiones antes de procesar cualquier trabajo. La especificación de la partición la puede designar el operador del computador o se puede incorporar en el sistema operativo. Cada paso de trabajo proporcionado por un usuario debe especificar la máxima cantidad de memoria necesaria; de esta forma encuentra y asigna una partición de tamaño suficiente. La técnica de especificación estática es especialmente apropiada cuando se conocen bien los tamaños y frecuencia de los trabajos; en esos casos se escogen los tamaños de las particiones para que correspondan lo mejor posible a los tamaños de trabajos más comunes, aunque se puede desperdiciar mucha memoria si no se conocen los tamaños y frecuencias de los trabajos.

2) Especificación de partición dinámica.

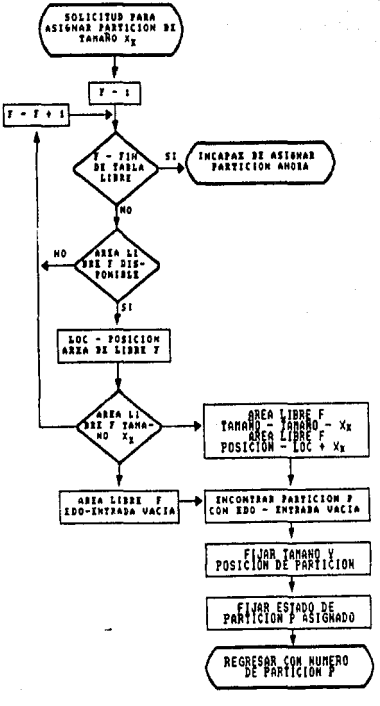
En la especificación dinámica las particiones se crean durante el procesamiento del trabajo para que se hagan corresponder los tamaños de las particiones con los de los trabajos.

Para la asignación primero se debe encontrar un área al menos de la misma magnitud que la partición deseada; luego, si el área es mayor que lo necesario, se debe partir en dos fracciones (una se convierte en la partición asignada y la otra en un área libre más pequeña). Recíprocamente, cuando se desasigna una partición, se trata de combinarla con cualquier área libre adyacente para formar un área libre contigua en vez de tener muchos pequeños pedazos.

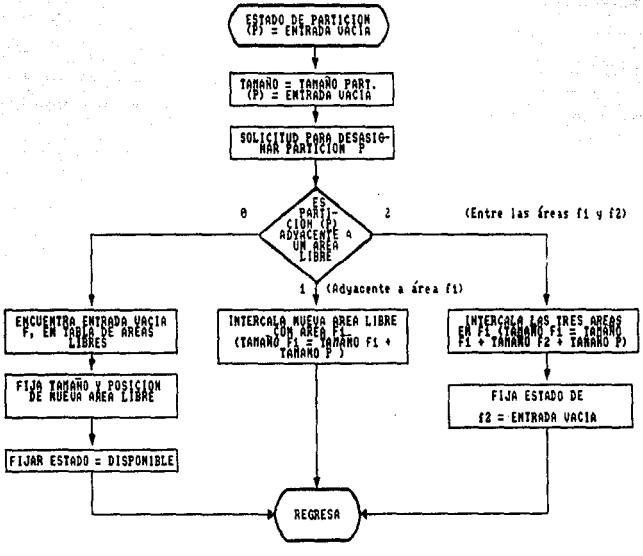
- Asignación y Desasignación.

En este método se dispone de varios algoritmos para efectuar funciones de asignación y desasignación. Se presentan dos ejemplos.

Algoritmo de asignación de una partición (figura 2.7)



Algoritmo de desasignación de partición (figura 2.8)



Existen dos variaciones a los algoritmos presentados. Estan se denominan primer ajuste y mejor ajuste.

a) Algoritmo de partición del primer ajuste:

En este método cuando se necesita asignar una partición, se comienza por el área libre en la dirección más baja de memoria y se prosigue la búsqueda hasta encontrar la primer área libre suficientemente grande para que ajuste la partición.

Ventajas:

- Generalmente se puede saber si la partición es adyacente a algún área libre.
- La técnica consiste en el empleo de áreas libres en la dirección interior de la memoria siempre que es posible. Lo que permite formar un área libre grande en las direcciones superiores de la memoria por lo que cuando se requiere una partición de gran tamaño, existe una gran probabilidad de encontrar un área libre de tamaño suficiente.

b) Algoritmo de partición del mejor ajuste:

La única diferencia entre esta técnica y la anterior es que las áreas libres deben estar clasificadas según su tamaño, es decir la primera entrada correspondera al área libre que tiene el tamaño más pequeño. En consecuencia el mejor ajuste es la primer área libre encontrada de tamaño suficiente para la partición deseada.

Ventajas:

- Si hay un área libre de exactamente el tamaño deseado se selecciona (lo que no necesariamente ocurre en la técnica anterior).
- Si no hay un área libre del tamaño exacto deseado, se obtiene la partición del área libre más pequeña posible sin destruir un área libre grande que puede necesitarse para satisfacer una solicitud posterior de una partición grande.

Desventajas:

- Generalmente el área libre no tiene el tamaño exacto necesario y por tanto debe dividirse en dos partes. Debido al criterio de la técnica, el área libre resultante regularmente es bastante pequeña, tanto que muchas veces es inservible.

En resumen el mejor ajuste tiende a acumular un gran número de áreas libres muy pequeñas y solo un área libre grande. Para dar por concluido el estudio acerca de la memoria particionada se listarán las desventajas que el uso de este método implica:

- La fragmentación puede constituir como se mencionó anteriormente un problema importante. Ya que es posible encontrar secuencias de trabajo que produzcan una utilización de memoria bastante inferior al 10 por ciento. El grado de este problema depende de las secuencias de trabajo y algoritmos utilizados.
- Aunque no se fragmente la memoria, el área libre aislada puede no ser suficiente para una partición. Por ejemplo, se tiene una memoria de 512K y una secuencia de trabajos que contenga trabajos de 300K y 256K. Si se asigna una partición de 300K, se desperdician los 212K restantes ya que éstos son insuficientes para cualquiera de los otros trabajos.
- La memoria puede contener información que nunca se use. Más aún, el tamaño de la partición de un trabajo está limitado al tamaño de la memoria física.

2.3 Administración de memoria por particiones relocalizables

La idea principal de este método consiste en mover celdas de memoria de un lugar a otro para tener áreas libres en lugares contiguos, lo cual es una solución al problema de fragmentación y permite tener un mayor grado de multiprogramación, que a su vez significa mayor utilización de memoria y procesador. Las celdas no se mueven, sino que sus contenidos se copian de un lugar a otro, y aunque esto da origen a un nuevo problema (el de relocalización), permite mayor flexibilidad que la administración de memoria particionada. Pero resulta más costoso, ya que hay que compactar (mover) los procesos al mismo tiempo de ejecución y realiza algunos cambios en el procesador central, para evitar que este desplazamiento cause problemas con respecto a las direcciones. Aunque en principio es simple, el mover una partición no garantiza que el trabajo continúe ejecutándose bien, en su nueva localización, debido principalmente a que pueden existir puntos sensibles a la posición tales como:

- Registros bases.
- Instrucciones de referencia a la memoria.
- Listas de parametros que utilizan indicadores de direcciones.

Para operar en forma adecuada este método se debe tener cuidado en modificar todos los puntos anteriores en el momento de asignar la nueva localización. Una solución al problema de reubicación es recargar todos los trabajos por reubicar y reiniciarlos desde el principio.

El cargador de reubicación puede manejar la reubicación estática inicial. Sin embargo este reinicio puede no ser factible en los casos que el programa haya efectuado acciones irreversibles.

Debido a lo anterior se han desarrollado varias técnicas de hardware para resolver el problema de reubicación de las cuales la mayoría de ellas están basadas en el concepto de un mapeo de memoria, en el que se distingue entre el espacio de dirección lógica visto por un espacio ejecutante y el espacio de dirección física de la memoria. En cada referencia de memoria el hardware mapea la dirección lógica suministrada por el programa a una dirección física correspondiente.

Este mapeo puede efectuarse en cierto número de formas a fin de cumplir las necesidades antes planteadas. Es deseable que el mapeo de dirección siempre sea eficaz cuando el procesador está corriendo en modo normal, pero capaz de ser rehabilitado por programas que corran en modo de sistema. Existen dos enfoques comunes al problema de la reubicación los cuales requieren de apoyo especial del hardware:

- **Tipificación de datos.**
- **Reubicación dinámica.**

El mapeo de direcciones mediante la tipificación de datos registra físicamente el tipo de valor almacenado en cada posición de memoria. Esto es posible debido a que el programa del usuario tiene un enfoque especial en lo que respecta a los indicadores de direcciones que no se pueden examinar o manejar arbitrariamente; en consecuencia, el usuario no puede percatarse de la posición porque no hay manera de que el programa determine su posición. Por ejemplo se agregaran dos bits a cada palabra para designar el tipo de valor 00 = entero, 01 = numero con punto flotante, 10 = carácter, 11 = indicador de dirección. .

El programa de usuario jamás utilizará los bits que identifican el tipo aunque el hardware los coloca automaticamente. Suponga que el enunciado es A = B ya que no sólo hace el valor de A igual a B, sino que también se copia el tipo de información, y se pueden manejar y mover los indicadores de dirección dentro del programa.

Cuando es necesario compactar y reubicar al sistema operativo usando instrucciones de alto nivel, se puede examinar la información de la clave de tipo y con ello encontrar cada indicador de direcciones que debe modificar. El enfoque de tipificación de datos tiene varias desventajas entre ellas:

- Requiere de bits adicionales en cada palabra.
- La compactación puede ser lenta ya que se deben de examinar los bits de tipo de cada palabra.

En la técnica de reubicación dinámica existen dos registros privilegiados especiales:

El registro base de reubicación (relocalización) y el registro de fronteras que solo puede accesar el sistema operativo. En cada referencia de memoria, automáticamente se suma el contenido del registro base de reubicación el cual se realiza conforme se ejecuta cada instrucción. La pregunta ahora es ¿ Cuándo debe de efectuarse la compactación? Para responder esta cuestión existen dos alternativas de las cuales la primera es compactar inmediatamente después que se desasigne una partición para quegiempre exista una sola área libre contigua sin fragmentación. El mantenimiento de la tabla de áreas libres y la asignación de particiones podría ser muy simple, sin embargo la compactación puede ser una operación demasiado lenta. La segunda alternativa es operar exactamente con los algoritmos de partición no reubicable compactando la memoria siempre que no se tuviera un área libre del tamaño suficiente. La compactación ocurriría con mayor frecuencia que con el plan mencionado, aunque el mantenimiento de la tabla sería mucho más complejo. Desventajas de la reubicación dinámica:

- El hardware de reubicación aumenta el costo de la computadora y puede bajar la velocidad.
- El tiempo de compactación puede ser considerable.
- Se deja de usar parte de la memoria porque aunque haya sido compactada, la cantidad de área libre puede ser menor que el tamaño necesario de partición. El espacio desperdiciado es aproximadamente igual a la mitad del tamaño de partición del trabajo promedio.
- La memoria puede contener información que nunca se use, sin embargo el tamaño de la partición está limitado al espacio de memoria contigua disponible.

Memoria Virtual

Cuando los programadores se enfrentaron por primera vez con programas demasiado grandes para tenerlos en memoria, la solución solía ser dividir los programas en partes, llamadas superposiciones. La superposición se ejecutaba y al terminar se llamaba a la siguiente. Algunos de esos sistemas de superposición eran demasiado complejos y admitían varias superposiciones en la memoria al mismo tiempo. Las superposiciones se guardaban en el disco y se intercambiaban en la memoria por parte del sistema operativo .

El trabajo de la división del programa en partes era realizado por el programador y éste consumía mucho tiempo. En 1961 *Fotheringham* creó una memoria que hoy se conoce como memoria virtual, que es un acceso a memoria principal y memoria secundaria indistintamente como si fueran idénticas. La idea básica que implica la memoria virtual es que el tamaño del programa, los datos y la pila combinados pueden exceder la cantidad de memoria física disponible para él. El sistema operativo guarda aquellas partes del programa que se encuentran en uso corriente en la memoria central y el resto en el disco. El manejo de memoria virtual exige un tratamiento de las direcciones para averiguar si se encuentran en la memoria principal o no; en caso de que no estén, el sistema operativo coloca el dato indicado en un lugar de memoria principal.

Se pueden distinguir tres tipos de memoria virtual:

a) Paginada.
b) Segmentada.
c) Segmentada-Paginada.

2.2.4 Administración de memoria paginada.

Esta técnica consiste en dividir la memoria en cuadros de página de tamaño físico. El espacio lógico de dirección visto por el usuario se encuentra dividido en páginas del mismo tamaño que los cuadros de página; una dirección generada por un programa se interpreta como un número de página y un desplazamiento dentro de la misma. Para lograr el mapeo sobre las tareas deseadas, una tabla de páginas mapea los números de página sobre los marcos de página.

Características:

1) Lleva el control del estado por medio de dos conjuntos de tablas:

   a) Tablas de mapas de página por espacio de dirección y cada una contiene una entrada por cada página.
   b) Tablas de bloques de memoria: una en el sistema, que contiene una entrada para cada bloque de memoria con información relativa al uso de ese bloque.

2) El planeador de trabajos determina primordialmente *quién* obtiene la memoria.

3) En la asignación se cargan todas las páginas del trabajo a bloques asignados, y se realizan las entradas apropiadas en la tabla de bloques de la memoria.

4) Para la desasignación, al terminar el trabajo se colocan los bloques en estado libre ajustando las entradas a la tabla de bloques. En un sistema paginado cada proceso ocupa cierto número de páginas en el espacio virtual de memoria. La cantidad total de memoria virtual empleada por los procesos conocidos por el sistema excederá en mucho a la memoria física disponible, aun cuando cada proceso requiera sólo la fracción de su memoria virtual que está disponible en cualquier instante. Por consiguiente, se considera que cada proceso reside normalmente en disco y la función del manejo de memoria es traer páginas de procesos activos a la memoria principal según se necesite. Cuando más cuadros de página se asignan a un proceso, es más probable que se encuentre la página referida para un acceso dado en la memoria principal.

Cuando un proceso intenta el acceso a una página que no se encuentra en la memoria física se genera una interrupción por falla de página. El manejador de interrupciones activa las rutinas de manejo de memoria que detienen el proceso que efectúa la solicitud en un semáforo e introducen una solicitud para que la página se recupere del disco. El semáforo pasa como parámetro de manera que el proceso detenido pueda ser liberado cuando la página esté disponible. Si hay cuadros de páginas en memoria disponibles la operación es directa, pero lo normal es que toda memoria física está ocupada, de tal forma que es necesario seleccionar la página que se sacara; esta selección corresponde al algoritmo de cambio de página o algoritmo de reemplazo. El trabajo del algoritmo de reemplazo consiste en seleccionar un cuadro de página que deba utilizarse para acomodar la página de memoria virtual. En la primera etapa del algoritmo se deben de identificar los candidatos para efectuar la eliminación; para esto existen dos estrategias: reemplazo por proceso y reemplazo global. La primera estrategia garantiza que cuando un proceso necesita otro cuadro de página, una de sus propias páginasse elegirá para reemplazo. Si se utiliza

esta estrategia, cada proceso activo tiene asignada una cuota de memoria real; las solicitudes de página se cubren con esta cuota cuando se agota, las solicitudes de páginas dan como resultado que se sobreescriba en una página existente. La desventaja del reemplazo por proceso es que la elección de una cuota adecuada es difícil y una cuota fija no toma en cuenta el tamaño variable del conjunto de trabajo. El resultado es que algunos procesos terminan con más memoria real de la que necesitan, mientras que otros no pueden acomodar el conjunto de trabajo dentro de su cuota.

En el reemplazo global todas las páginas existentes en la memoria real excepto las páginas del sistema aseguradas son candidatas a reemplazo. Esta estrategia es más simple de administrar pero está expuesta a críticas ya que el avance de un proceso particular no está determinado tan sólo por su propio patrón de accesos a la memoria, sino por una interacción compleja entre los patrones de acceso a la memoria en todos los procesos activos del sistema.

Desventajas:

1. Generalmente el hardware de mapeo de direcciones de páginas aumenta el costo de la computadora.
2. Se debe usar memoria para reservar las distintas tablas y tiempo del procesador para mantener y actualizar las mismas.
3. Aunque este método elimina la fragmentación, llega a ocurrir un fenómeno conocido como fragmentación interna o ruptura de páginas se asignan bloques completos de memoria a los trabajos. Por ejemplo si un trabajo requiere 5K y el tamaño de bloque es de 4K, se deben asignar dos bloques (se asignan 8K aunque se requieran 5K). Debido a esto en promedio se desperdicia media página para cada trabajo. Se debe de llegar a un arreglo entre reducir el desperdicio debido a la ruptura de páginas, lo que favorece las páginas pequeñas, y reduce el número de entradas en las tablas de mapas de las páginas. Para poder resolver este problema es importante conocer el tamaño promedio de los trabajos.
4. Queda sin utilizar parte de la memoria si el número de bloques disponibles no es suficiente para los espacios de direcciones de los trabajos por correr.
5. El espacio de dirección de un trabajo está limitado al tamaño de la memoria física.
6. La memoria contiene información que rara vez se utilizara.

2.2.5 Administración de memoria paginada por demanda.

Cualquiera que haya sido la estrategia adoptada, en la técnica de administración paginada respecto al algoritmo de reemplazo,la siguiente decisión que debe tomar es respecto al intentar anticipar las necesidades de paginación o dejar que la paginación esté manejada por eventos recuperando páginas según se necesite, esta técnica se conoce como paginación de memoria por demanda.

Ventajas:

1. Memoria virtual grande: El tamaño físico de la memoria ya no limita el espacio de dirección de un trabajo, lo que proporciona mejor la compatibilidad entre las computadoras grandes y pequeñas.
2. Uso más eficiente de la memoria: No se mantienen en la memoria física las porciones del espacio de dirección de un trabajo que no se usan o que se usan rara vez.
3. En este método el grado de multiprogramación no se limita por efectos de memoria física.

Cuando ocurre una falla de página, el sistema operativo tiene que escoger una página para retirarla de la memoria y dejar espacio para la página que tiene que traerse. Si la página que va ser sustituida se ha modificado en su estancia en la memoria, ésta debe reescribirse en el disco para actualizar la copia en el mismo, de lo contrario, es decir, si la página no fué modificada no es necesario volverla a escribir.

El rendimiento del sistema es óptimo si se elige una página que no se ha utilizado demasiado, ya que si se retira una página bastante utilizada, probablemente tendrá que devolverse con rapidez, lo cual producirá costo adicional. Sobre algoritmos de cambio de página se han realizado muchos trabajos tanto teóricos como experimentales de entre los cuales se presentan los siguientes:

a) Sustitución óptima de páginas.

Al momento en que ocurre la falla de una página, un conjunto de páginas está en la memoria, una de estas páginas será referida en la siguiente instrucción. Cada página puede rotularse con el número de instrucciones que se ejecutaran antes de que la página se refiera por primera vez. Este algoritmo asegura que la página con el rótulo mayor debe eliminarse. Cuando ocurre la falla de página, el sistema operativo no tiene manera de saber cuál de las páginas se llamara después. A pesar de ello, por medio de la ejecución de un rograma en un simulador y el control de todas las referencias hechas a páginas, es posible implementar la sustitución de página óptima en la segunda ejecución utilizando la información de referencias a páginas que se recolectaron en la primera ejecución.

b) Sustitución de página no usada recientemente.

Para que el sistema operativo colecte estadísticas utiles acerca de qué páginas se están utilizando y cuales no, la mayoría de las computadoras con memoria virtual tienen dos bits asociados con cada página. Un bit, el bit R o referido, es colocado por el hardware en cualquier lectura o escritura en la página. El otro bit, el bit M o modificado, lo forma el hardware cuando se escribe la pagina. Es importante comprender que estos bits se deben actualizar en cada referencia de la memoria, de manera que es esencial que sean fijados por el hardware. Una vez que un bit se haya fijado en 1, este seguira siendo 1 hasta que el sistema operativo lo devuelva a 0 en software.

Si el hardware no tiene bits R y M, éstos pueden simularse de la siguiente forma: Cuando un proceso se inicia, todas sus captaciones de la tabla de páginas se marcan como ausentes en la memoria. Tan pronto como se haga refencia a cualquier página ocurrirá una falla de página. Después el sistema operativo fija en sus tablas internas el bit R, cambia la captación de la tabla de páginas para que apunte a la página correcta, con el modo sólo de lectura y reinicia la instrucción. Si después se reescribe la página

ocurrirá otra falla de página, permitiendo que el sistema operativo fije el bit M y cambie el modo de la página por lectura/escritura.

c) Sustitución de páginas donde la primera entrada es la primera salida (FIFO).

En este algoritmo el sistema operativo conserva una lista de todas las páginas que están actualmente en la memoria, donde la página que está a la cabeza de la lista es la más antigua y la de la cola es la de más reciente ingreso. Al fallar la página, la página que está a la cabeza se retira y se anexa la nueva en la cola de la lista. Una modificación al algoritmo que evita el problema de desechar una página de uso frecuente consiste en inspeccionar los bits R y M de la página más antigua.

Otra variación de este método es la segunda oportunidad la cual consiste primero en inspeccionar la página más antigua. Si su bit R es 0, la página se sustituye de inmediato. Si R es 1, el bit se elimina y la página se coloca al final de la lista de páginas, como si acabara de llegar a la memoria. Después continúa la búsqueda. Lo que se hace en esta segunda oportunidad es buscar una página antigua que no haya sido referida en el intervalo anterior del reloj. Si se han referido todas las páginas, la segunda oportunidad degenera en lo expuesto primeramente.

d) Sustitución de página usada menos recientemente.

Este algoritmo tiene el siguiente principio: " cuando ocurra una falla de página, deséchese la página que haya estado sin uso por el periodo de tiempo más largo ". Para implementar completamente esta estrategia, se necesita conservar una lista enlazada de todas las páginas que están en la memoria, con la más reciente en el frente y la menos usada en el fondo. La dificultad que esto presenta es que la lista debe actualizarse en cada referencia a la memoria. La localización de una página en la lista, su supresión y después su traslado al frente es una operación que se lleva mucho tiempo y que además es muy costosa ya que se necesita hardware especial.

Además de las desventajas mencionadas para la administración paginada tales como costos de hardware, ruptura de páginas, alto del procesador y espacio de memoria para las tablas se tiene lo siguiente: El número de tablas y la cantidad del alto del procesador para manejar las interrupciones de páginas son mayores que en el caso de la administración paginada simple.

2.2.6 Administración de memoria segmentada:

La segmentación proporciona un sistema de direccionamiento bidimensional en memoria virtual; de esta forma una dirección virtual está constituida por dos partes, un especificador de segmento y la ubicación dentro del mismo. Se puede definir un segmento como una agrupación lógica de la información, tal como una subrutina, arreglo o área de datos; en consecuencia cada espacio de dirección de un trabajo consiste físicamente en una colección de segmentos. A diferencia de las páginas, que son de tamaño fijo, los segmentos pueden ser de longitud arbitraria de acuerdo con el problema que se está tratando.

Ventajas:

1. Elimina la fragmentación. Al mover los segmentos, se puede combinar el espacio de memoria fragmentada en una sola área libre.
2. Proporciona memoria virtual. Al mantener en memoria principal sólo los segmentos que son usados activamente, el tamaño del espacio total de la dirección del trabajo puede exceder al de la memoria física.
3. Admite segmentos dinámicamente crecientes o la comprobación automática de fronteras. Por ejemplo, si se necesita aumentar el tamaño de un segmento durante la ejecución, se puede detectar una referencia "fuera de rango" mediante el componente de tamaño de cada entrada en la tabla de mapa de segmentos.

4. Facilita segmentos compartidos (áreas de datos y procedimientos). Si dos trabajos utilizan una misma rutina *determinada*, es un desperdicio tener dos copias por separado en la memoria principal.
5. Impone el acceso controlado. Se debe controlar el acceso a cada segmento, esto es una tabla de constantes debe quedar restringida sólo a acceso de lectura, tanto que un segmento de área de trabajo puede ser escrito y leído. En la memoria segmentada se mantiene una tabla de *segmentos* para cada proceso que contiene por cada segmento conocido para el proceso, la dirección virtual del inicio del segmento y su tamaño. Un registro de procesador apunta a la tabla de segmentos para el proceso que se está ejecutando en ese momento. La primera etapa en la traducción de dirección es utilizar la tabla de segmentos para verificar el desplazamiento y después generar la dirección virtual del dato requerida. El resultado de la etapa de traducción sigue siendo una dirección virtual. En la mayor parte de los sistemas segmentados, los segmentos mismos se paginan para evitar la fragmentación de la memoria física.

Las funciones de la administración de memoria segmentada combinan algunos de los aspectos de la administración de memoria por particiones relocalizables y paginada por demanda.

1. Lleva el control de estado a través de tres conjuntos de tablas.

   a) Tablas de mapas de segmentos: una por espacio de dirección.
   b) Tabla de área no asignada: una en el sistema.
   c) Tabla de nombres de segmentos: una en el sistema.

2. El planeador de trabajos puede determinar estáticamente la política de quién y cuándo recibe la memoria asignada, si el uso de la memoria virtual está limitado es decir, el tamaño de la memoria virtual es comparable con el de la memoria física.

3. Cuando se debe de asignar un segmento, se debe encontrar un área disponible suficientemente grande; de ser necesario debe efectuarse la compactación.

4. Cuando se desea asignar un segmento y no es posible encontrar un area suficientemente grande para esto, se debe desasignar uno o más de los segmentos asignados hasta ese momento. La memoria liberada queda como espacio disponible. Al concluir un trabajo, toda la memoria utilizada por segmentos queda disponible. La protección al nivel del segmento generalmente es mucho más deseable que proteger las páginas o bloques de memoria, debido a que los segmentos representan entidades lógicas significativas al programador, por lo cual el mecanismo de mapeo de dirección debe verificar que se permita el acceso. Ya que es posible sacar segmentos de la memoria principal y enviar a almacenaje secundario, es necesario indicar si el segmento se encuentra en ese momento en la memoria. En caso contrario, debe ocurrir una interrupción al sistema operativo. Para ayudar al sistema operativo a decidir cuáles segmentos mantener en memoria y cuales sacar, se pueden utilizar bits referenciados y cambiarlos en cada entrada de la tabla del mapa e segmentos.

Desventajas:

1) El manejo de memoria segmentada incrementa el costo del hardware, necesita espacio adicional para las tablas y aumenta la complejidad del sistema operativo.
2) El elevado uso de compactación como soporte al crecimiento dinámico de *segmentos* para evitar la fragmentación.
3) Existe dificultad en el manejo de segmentos de tamaño variable en el almacenamiento secundario.

2.2.7 Administración de memoria paginada y segmentada.

Un camino para obtener el beneficio de la segmentación y eliminar muchas de sus desventajas es combinar los mecanismos de paginación y segmentación. En vez de tratar a cada segmento como una simple entidad contigua, cada uno puede ser subdividido en páginas. Con la manipulación física de esas páginas desaparecen los problemas de compactación, almacenamiento secundario y la limitación del tamaño del *segmento*. Al *cargarse* las *páginas* en memoria real se colocan en los bloques que el sistema operativo ha asignado. Los segmentos se cargan de forma que una página entre en un bloque.

Estas páginas pueden colocarse en cualquier bloque de memoria que está libre. Conforme se realiza la carga, se crean tablas para identificar o representar las posiciones de segmento y de página en memoria real, las cuales se emplearán para la traducción dinámica de direcciones *durante* la *ejecución* del programa. Los sistemas con segmentación y paginación utilizan una tabla de segmentos y una tabla de páginas por cada *segmento* que haya en el espacio de direcciones de un programa. La tabla de páginas indica al sistema operativo dónde está situada una página en la memoria real durante la traducción dinámica de direcciones. La tabla de páginas propiamente dicha está en la memoria real y cada segmento de un programa posee una tabla de páginas. La computadora encuentra la tabla de páginas mediante una entrada en la tabla de segmentos del programa ya que ada segmento posee su propia tabla de páginas.

La tabla de segmentos también esta localizada en la memoria. El dispositivo de traducción dinámica de direcciones de la computadora precisa el registro de origen de la tabla de segmentos y las tablas de páginas. Si la computadora está funcionando en el modo de multiprogramación, entonces el sistema operativo ha de ser capaz de localizar una instrucción de cualquier programa en cualquier instante. Esto quiere decir que tendra que construir otra tabla que contenga los registros de origen de la tabla de segmentos de cada programa. Esta tabla tendra dos entradas verticales: una indica el nombre del programa y otra, la dirección de comienzo de la tabla de *segmentos* del programa.

La traducción dinámica de direcciones la efectua automaticamente un dispositivo físico de la computadora denominado DAT. En los sistemas de segmentación y paginación, la traducción tiene lugar durante toda la ejecución del programa. La segmentación es el mejor método para la gestión del espacio de direcciones de un programa o de un sistema. La paginación es un buen procedimiento para el manejo de la memoria real de un sistema y da como resultado un mínimo de pérdidas ocasionadas por la fragmentación. La segmentación ofrece varias ventajas adicionales, como la protección de segmentos. La combinación de la segmentación y la paginación produce más beneficios a los usuarios del sistema que la sola utilización de la segmentación o de la paginación. La implementacion de la segmentación y la paginación en un sistema de multiprogramación puede hacerse de dos formas:

1. Las tablas de segmentos y páginas que están en la parte del sistema operativo.
2. Cada programa que se procese en el sistema puede tener su propia tabla de *segmentos* y de páginas.

Matrix de registros asociativos.

Si en la ejecución de un programa se utiliza el metodo de traducción por tablas de segmentos y páginas, dichos programas se ejecutarán muy lentamente. Durante la traducción, el dispositivo DAT de la CPU debe hacer referencia a las tablas de segmentos y de *páginas*, y estas tablas residen en la memoria principal. La velocidad de la memoria real será lenta si se compara con la velocidad de la CPU.

Aparentemente las ventajas de este método tienen su contrapartida en esta reducción de la velocidad de ejecución, lo cual sería verdad si no fuera por cierto equipo físico especial denominado matriz de registros, de que disponen los sistemas donde puede ser implementada este tipo de administración de memoria. Esta matriz de registros es un dispositivo de finalidad especial, y como su nombre lo indica es un conjunto de registros los cuales son mucho más rápidos que la *memoria principal*; sirven para compensar la velocidad relativamente baja de la traducción con tablas. Las computadoras grandes tienen varios K de estos *registros*. *Supongase* que se han asignado ocho registros a la ejecución de un programa.

Imagínese que se ejecuta una instrucción que hace referencia a una posición de memoria; la cual viene indicada por el *número de segmento*, el de página y el de desplazamiento dentro de la página. Para acceder a esa posición se compara el contenido de la dirección relativa con la primera parte de los registros. Los registros contienen ocho páginas del programa a las que últimamente se hicieron referencia, páginas identificadas por sus números de segmento y de página. Si existe una coincidencia con una de las entradas de cualquier registro, la correspondiente posición de bloque se enlaza con el desplazamiento de la dirección relativa; lo que da como resultado la *dirección* de la memoria real que se solicita. De igual manera, el bit de referencia de registro se pone en ON. Las ocho comparaciones se producen al mismo tiempo y en paralelo ya que la matriz de registros asociativos es muy rápida y el tiempo de traducción es insignificante. En caso de no existir coincidencia entre la dirección relativa y los registros de la matriz asociativa, la traducción se completa con tablas al mismo tiempo que en la matriz asociada. Si en alguno de los registros no hay coincidencia, la traducción se completa con las tablas; posteriormente el número del bloque resultante, los números de segmentos y de página relacionados con ella se colocarán en uno de los registros de la matriz asociativa. El registro seleccionado se determina utilizando los bits de referencia.

2.3 Manejo de Información

El manejo de la información se refiere al almacenaje y recuperación de la *información* confiada al sistema. Sus funciones son las siguientes:

1. Llevar el control de toda la información dentro del sistema mediante varias tablas, la principal que se conoce como tabla de contenido del volumen o directorio de archivos. Estas tablas contienen el nombre, localización y derechos de acceso de toda la información dentro del sistema.
2. Decidir la política para determinar como y dónde se reserva la información y quién tiene acceso a ella. Algunos factores que influyen en esta política son la utilización eficiente de memoria secundaria, acceso eficiente, flexibilidad a usuarios y la protección de los derechos de acceso a la información solicitada.
3. Asignar el recurso de información, después de haber tomado la decisión de dejar que un proceso tenga acceso a la información, los módulos de asignación deben encontrar la información deseada, hacer accesible dicha información al proceso y determinar los derechos propios de acceso.
4. Desasignar el recurso. Cuando ya no se necesita información, las entradas temporales de las tablas y otros *recursos* semejantes se pueden liberar. Si el usuario ha actualizado la información se puede también actualizar el archivo original.

Los módulos de la administración de información son:

- Manejo de Archivos.
- Manejo de trabajos.

2.3.1 Manejo de Archivos.

Las funciones básicas del manejo de archivos son:

1. Llevar control de la información dentro del sistema mediante varias tablas.
2. Decidir la política para determinar cómo y dónde se reserva la información y quién tiene acceso a ella.
3. Escoger el periférico que contiene la información.
   - Encontrar la información deseada, hacer posible el acceso del proceso de la misma y determinar los derechos propios de acceso.
4. Designar el recurso. Si el usuario ha hecho las operaciones correspondientes y no va a hacer más, se puede actualizar el archivo original de esa información para el uso posible

de otros procesos. Los módulos del manejo de datos se conocen, a veces, como sistema de archivos, el cual normalmente contiene:

- Métodos de acceso: Estos son los que se relacionan con la forma en la cual se accesan los datos almacenados en los archivos.
- Administración de archivos: Es la parte relacionada con los mecanismos de almacenamiento, distribución y seguridad de los archivos.
- Auxiliar de la administración de almacenamiento: se encarga de asignar un espacio libre para los archivos en dispositivos de almacenamiento secundario.
- Archivo de integridad: Garantiza que la información en un archivo no se modifique.

Normalmente a un sistema de archivos se le atribuyen las siguientes características:

a) El usuario:

- Debe poder crear, modificar y borrar archivos.
- Tiene la posibilidad de distribuir cada uno de los archivos de manera ordenada y controlada según las necesidades del mismo.
- Puede ordenar transferencias de información entre archivos.
- La capacidad de respaldo y recuperación debe ser considerada para prever pérdidas accidentales o destrucción maliciosa de información.

b) El mecanismo para distribuir archivos debe proveer varios tipos de acceso controlado tales como lectura, escritura, acceso de ejecución o varias combinaciones de los mismos.

c) En determinados ambientes en los cuales la información debe guardarse en forma segura y privada, los archivos pueden proveerse de técnicas de encriptación y desencriptación.

d) El sistema de archivos debe proveer al usuario una interfaz agradable. Lo cual dara al usuario una vista lógica de sus datos y las funciones para una representación física de los mismos.

Se pretende que el sistema de archivos proporcione la gestión conveniente, tanto de información permanente como temporal.

Un objetivo del mismo es permitir al programador preocuparse solamente por la estructura lógica de su programa y por las operaciones realizadas en el proceso de su información. Los sistemas de archivos también pueden permitir compartir la información entre los usuarios y proteger la información del acceso no autorizado.

Organización de Archivos.

La organización de archivos se refiere a la manera en la cual los registros de un archivo son ordenados en almacenamiento secundario. Las más populares técnicas de organización utilizadas actualmente son las siguientes:

- Secuencial: Los registros se almacenan uno a continuación del otro en orden marcado por un campo llamado llave. Esta organización es usada para archivos sobre cinta magnética, cinta de papel, tarjetas perforadas o discos.
- Secuencial indexada: Esta organización da una capacidad para preparar procesos secuenciales de acceso directo o random a través del uso de índices, los cuales se emplean para señalar o apuntar a los registros que están almacenados en un archivo. Los registros son ordenados en una secuencia lógica de acuerdo a la llave que contiene cada uno de ellos. El sistema mantiene un índice que contiene las direcciones físicas de los principales

registros. Los registros secuenciales *indexados* pueden ser accesados secuencialmente mediante una llave o directamente mediante una búsqueda a través del sistema de índices creado. Los archivos secuenciales indexados son normalmente almacenados en disco.

- Directo: Estos archivos sólo podrán ser generados en dispositivos de almacenamiento directo debido a que estos archivos tienen la capacidad de tener acceso aleatorio a los registros. Con la organización directa se genera una dirección en el dispositivo de acceso a partir de una llave *dentro* del *registro* del archivo. Esta llave en el dispositivo es única y permite al sistema especificar la dirección del registro y así este podrá ser accesado inmediatamente, sin tener que procesar otro registro del archivo.
- Particionada: Esta es *esencialmente* un archivo de subarchivos secuenciales; cada uno de ellos es llamado miembro. El inicio de la dirección de cada miembro es almacenado en el archivo de directorio. Este método es usualmente utilizado para almacenar programas de librerías o macro librerías.

Técnicas de asignación del espacio libre.

Los archivos se crean y borran frecuentemente; como sólo hay una cantidad limitada de espacio en disco, es necesario reutilizar el espacio de los archivos borrados para los nuevos archivos. Para conocer el espacio libre en *disco*, el *sistema* de archivos mantiene una lista de espacio libre. Esta lista registra todos los sectores del disco que se encuentran libres; al crear un archivo, se examina la lista de espacio libre para checar si existe la cantidad precisa de espacio y asignarlo a un *espacio* libre. Este espacio desaparece entonces de la lista de espacio libre, cuando se borra el archivo su espacio se añade a ésta. La lista de espacio libre se implementa como un vector de bits o mapa de bits. Cada sector se representa por un bit; si el sector está libre, el bit estará en 0, y 1 en caso contrario. Esto implica que todos los sectores han de tener un número de posición asignado. Otra técnica consiste en encadenar todos los sectores libres del disco, guardando un apuntador al primer sector libre; este sector contiene un apuntador al siguiente sector libre, y así en forma consecutiva hasta el *último* sector libre. Este esquema no es *muy eficiente*; ya que se debe recorrer toda la lista y leer cada sector, lo cual implica un tiempo mayor en operaciones de entrada/salida. Una modificación a esta técnica podría ser almacenar las direcciones de los n sectores libres en el primer sector libre. Los *primeros* n-1 sectores están libres y la última dirección es un apuntador a otro sector que contiene las direcciones de otros n sectores libres. La importancia de esta implementación es que las direcciones de un gran número de sectores libres pueden ser encontradas rápidamente.

La tercera técnica parte del hecho de que generalmente varios sectores contiguos están asignados o liberados simultáneamente, en particular, cuando se usa *asignación* contigua. De tal forma que es mejor guardar la dirección del primer sector libre y el número de n sectores contiguos libres que lo siguen a una lista de n direcciones libres.

MÉTODOS DE ASIGNACION DE MEMORIA:

a) Asignación contigua.

Los archivos son asignados a áreas contiguas de almacenamiento secundario (disco). Las direcciones de disco definen un ordenamiento lineal sobre el disco. El usuario especifica el tamaño del área necesaria para almacenar un archivo después de ser creado. Si la cantidad deseada de espacio contiguo no está disponible no se crea el archivo. Una ventaja de la asignación contigua es que los registros lógicos sucesivos son físicamente adyacentes uno a otro; lo cual provoca un acceso más rápido si se compara con sistemas en los cuales los registros lógicos están dispersos dentro del disco.

Para cada archivo con asignación contigua es necesario un registro de directorio para conservar la dirección del inicio y la longitud del archivo. La asignación contigua de un archivo se define por la dirección del disco del primer sector y su longitud. Si un archivo tiene n sectores (bloques) de longitud y comienza en la posición b: entonces este archivo ocupara los sectores b, b+1, ...,b+n-1. El acceso a un

registros lógicos sucesivos son físicamente adyacentes uno a otro; lo cual provoca un acceso más rápido si se compara con sistemas en los cuales los registros lógicos están dispersos dentro del disco.

Para cada archivo con asignación contigua es necesario un registro de directorio para conservar la dirección del inicio y la longitud del archivo. La asignación contigua de un archivo se define por la dirección del disco del primer sector y su longitud. Si un archivo tiene n sectores (bloques) de longitud y comienza en la posición b: entonces este archivo ocupara los sectores b, b+1, ...,b+n-1. El acceso a un archivo con asignación contigua soporta las dos formas de acceso más frecuentes: la directa y la secuencial.

Desventajas:

- Encontrar un espacio para un nuevo archivo es una dificultad para la asignación contigua. Una vez que se tiene la lista de espacio libre, se debe decidir cómo encontrar el espacio para asignarlo de forma contigua a un archivo. Si el archivo que se va a crear tiene n sectores de una longitud determinada, se debe buscar en la lista de espacio libre de n sectores contiguos libres. ó lo que es lo *mismo*, en el mapa de bits se necesita encontrar n bits contiguos con valor de 0.

El mayor problema para la asignación contigua es determinar cuánto espacio se necesita para un archivo, cuando el archivo se crea, la cantidad total de espacio que se necesitara debe de ser encontrada y asignada. La pregunta ahora es ¿ Cómo sabe el usuario el tamaño del archivo que se creara? en general, el tamaño del archivo de salida suele ser difícil de estimar. Si se asigna a un archivo grande un espacio muy pequeno, se puede encontrar con que no se puede redoclr el archivo. Para este problema existen dos soluciones:

1. El programa del usuario puede tomarse con un margen de error apropiado. El usuario debe asignar más espacio y ejecutar el programa de nuevo. Para prevenir estas rutinas repetidas las cuales son demasiado costosas, el usuario normalmente debe sobreestimar la cantidad de espacio necesitado, de ahí que halla un desperdicio considerable de memoria.

2. Entrar en un segmento no asignado (denominado hole) más grande, copiar el contenido del archivo al nuevo espacio y borrar el espacio anterior. Esta acción puede repetirse siempre y cuando exista espacio, aunque puede consumir bastante tiempo.

b) Asignación encadenada.

Con esta asignación cada archivo es una lista encadenada de sectores de disco. Soluciona la fragmentación externa y los problemas de declaraciones de tamaño de la asignación contigua. La asignación encadenada consiste en utilizar dos bytes en cada bloque como enlace para identificar al sucesor. Los bloques no utilizados se recolectan en una cadena desocupada: Se toman bloques de esta cadena para crear o ampliar un archivo, y cuando un archivo termina, sus bloques pueden devolverse a la cadena desocupada para volverse a utilizar. Con esta organización el archivo está identificado en forma única si se conoce el número de su primer bloque, de manera que la parte restante de la E/S lógica es un mapeo entre nombres simbólicos y números de bloque.

Con la asignación encadenada no hay fragmentación externa. El primer sector libre en la lista de espacio libre será usado para satisfacer una petición. No hay necesidad de indicar el tamaño de archivo que se crea. Un archivo puede seguir creciendo siempre que haya sectores libres; consecuentemente no es necesario compactar el espacio del disco.

Desventajas:

- El mayor problema de la asignación encadenada es que solamente se puede usar de forma efectiva para el acceso secuencial de archivos. Cada acceso a un apuntador requiere de una lectura de disco, por lo tanto, no se puede soportar el acceso directo para archivos con este tipo de asignación.
- Para poder encontrar un sector en un archivo se debe comenzar por el principio del mismo y recorrer los apuntadores en forma secuencial hasta que se encuentre el sector.
- El espacio requerido por los apuntadores es considerable. Otro problema con la asignación encadenada es la seguridad. Por ejemplo, considérese lo que sucedería si un apuntador se pierde o se estropea, esto provocaría un error en el software del sistema operativo o en el hardware del disco que puede dar origen a un puntero equivocado. Este error puede ocurrir en el encadenamiento de la lista de espacio libre o en el archivo.

c) Asignación Indexada.

La asignación indexada soluciona el principal problema de la asignación encadenada la cual no puede soportar el acceso directo de un archivo debido a que todos los sectores están desperdigados sobre el disco. Esto lo hace colocando todos los punteros juntos en el sector de índice.

Cada archivo tiene su propio sector de índice. Este sector es un vector de apuntadores (direcciones) a sectores, la i-ésima posición en el sector de índices apunta al i-ésimo sector del archivo. Para leer el i-ésimo sector, se usa el apuntador i-ésimo del sector de índice. Cuando se crea un archivo, todos los apuntadores en el sector de índices son puestos a NIL. Cuando el i-ésimo sector se va a escribir, se quita un sector de la lista de espacio libre y su dirección se pone en el i-ésimo puesto del sector de índices.

Una desventaja de la asignación indexada es que tiene una pérdida de espacio, suponga que se tiene uno o dos sectores por archivo. Con la asignación encadenada sólo se pierde el espacio de uno o dos apuntadores por sector. Con la asignación indexada se tiene un sector de índices del cual sólo se utilizan una o dos palabras para los apuntadores y el resto es memoria desperdiciada. La asignación indexada usa un sector de índice separado para apuntar a los sectores de índices, los cuales apuntan a los sectores del archivo. Para acceder a un sector, el sistema operativo usa el primer nivel de índice para encontrar el segundo nivel de índice, y este para encontrar el sector de datos deseado. Como cada archivo debe de tener un sector de índices éste debe de ser lo más pequeño posible; aunque si es demasiado pequeño no podrá contener suficientes apuntadores para un archivo grande, de ahí que un sector de índices sea normalmente un sector de disco. Y por consiguiente, puede leerse y escribirse directamente.

Sistemas de Directorios:

El directorio de un archivo es esencialmente una tabla de símbolos, cada usuario tiene un directorio que contiene una anotación por cada uno de sus archivos. Aún cuando el objetivo principal del directorio es mapear el nombre de archivo al número de bloque, la anotación de directorio contiene por lo general más información.

Contenido de una anotación de directorio de archivo

nombre del archivo
número del bloque inicial
control de acceso
(permiso escrito)
mantenimiento:
por lo general fecha/hora de creación, Fecha/hora del último acceso y longitud del archivo (en bloques)

El sistema operativo recibe el nombre simbólico del archivo y encuentra el archivo nombrado. Considerando una estructura de directorio particular, se debe tener en cuenta las siguientes operaciones que se ejecutan sobre un directorio:

1. Búsqueda. Se puede examinar una estructura de directorio para encontrar la entrada de un archivo en particular.
2. Crear un archivo. Pueden crearse y añadirse nuevos archivos al directorio.
3. Borrar un archivo. Cuando un archivo va a dejar de ser usado, hay que eliminarlo del directorio.
4. Listar un directorio. Se necesita ser capaz de listar los archivos y los contenidos de un directorio.
5. Copiar un archivo. Por seguridad, generalmente es bueno crear un respaldo para salvar el contenido y estructura de un archivo. Esto a menudo, consiste en copiar todos los archivos en una cinta magnética o disco.

En un sistema multiusuario existe un directorio de archivos por usuario, de manera que también puede haber un directorio de archivos maestro indexado por identificación de usuario, para dar la localización del directorio de usuario.

Sistema de directorios a dos niveles

Este sistema de directorios es un árbol de dos niveles: el primer nivel está formado por el directorio principal de dispositivo (sistema de archivo físico) y el segundo por los directorios de archivos (sistema de archivo lógico). El primero almacena información relativa a cada dispositivo físico y describe todos los archivos existentes en cada dispositivo. El segundo se concentra principalmente en describir las propiedades físicas de cada archivo: dónde está, qué longitud tiene, como está asignado. Los directorios de archivos se concentran sobre las propiedades lógicas de cada archivo. El directorio de archivos puede apuntar al directorio principal para proporcionar propiedades físicas o puede duplicar esta información.

En un directorio de dos niveles la raíz del árbol es el directorio maestro de archivo, sus descendientes directos son los directorios de los usuarios, los descendientes de éstos son los archivos. Estos archivos son las hojas del árbol. Al especificar un nombre de usuario y un nombre de archivo, se define un camino en el árbol desde la raíz hasta la hoja (el archivo especificado). Por tanto, un nombre de usuario y nombre del archivo definen un nombre de camino. Cada archivo en el sistema tiene un único camino. En una estructura de directorio dos niveles, cada usuario tiene su propio directorio de archivo de usuario. Cada directorio de usuario tiene una estructura semejante, pero lista sólo los archivos de un usuario.

Cuando un trabajo de usuario comienza, es examinado el directorio maestro de archivos. Este directorio es indexado por nombre de usuario o número de cuenta y cada entrada apunta a su respectivo directorio de usuario.

Cuando el usuario se refiere a un archivo en particular, sólo se examina el directorio de archivos del propio usuario. En una estructura de directorio a dos niveles, cada usuario tiene su propio directorio de archivo de usuario. Cada directorio de usuario tiene una estructura semejante al directorio maestro, pero lista sólo los archivos de un usuario. Por tanto usuarios diferentes pueden tener archivos con el mismo nombre, sin embargo los nombres de archivos dentro de cada directorio de usuarios deben ser únicos.

Esta estructura incomunica un usuario con otro, lo cual puede ser ventaja cuando los usuarios son completamente independientes, pero es lo contrario cuando éstos quieren cooperar en algunas tareas y acceder a archivos de otros usuarios.

Algunos sistemas no permiten que archivos locales de un usuario sean accesados por otros usuarios. Si el acceso se permitiera, el usuario debe de tener la posibilidad de nombrar a un archivo en otro directorio de usuario.

Sistemas de directorios estructurados en árbol.

La estructura de un directorio se puede extender a un árbol arbitrario. Esto permite a los usuarios crear sus propios subdirectorios y organizar sus archivos a su gusto. Esta estructuración se denomina también estructura de directorios jerárquica.

Un directorio estructurado en árbol proporciona un mapeo más elaborado entre el nombre del archivo y el archivo, y sólo cuando se ha identificado al archivo se necesitara acceso a la información de mantenimiento. De esta forma, se necesita una estructura de datos para que contenga esta información, entonces la estructura se vuelve una abstracción del archivo, y la función del directorio es mapear a partir de un nombre simbólico a esta estructura.

Un directorio o subdirectorio contiene un conjunto de archivos y/o subdirectorios. Todos los directorios tienen el mismo formato interno. Existe un bit en cada entrada del directorio que define la entrada como un archivo (0) o como un subdirectorio (1). Para crear o borrar subdirectorios se usan llamadas especiales al sistema. Algunos sistemas no borran un directorio a menos que éste se encuentre vacío. Por tanto, para borrar un directorio deben eliminarse primero todos los archivos de ese directorio. Si hay algunos subdirectorios, este procedimiento debe aplicarse a ellos recursivamente hasta que desaparezcan. En un sistema de directorio estructurado en árbol cada usuario tiene un subdirectorio actual. Este subdirectorio puede contener la mayoría de los archivos que son de interés para el usuario. Cuando se hace referencia a un archivo, el subdirectorio actual es revisado; si el archivo necesitado no se encuentra, entonces el usuario debe especificar el nombre del camino o cambiar el subdirectorio en uso. Los nombres de camino pueden ser de dos tipos: nombres de camino completo o nombres de camino relativo. Un nombre de camino completo comienza en la raíz y sigue un camino hacia el archivo especificando los nombres de los directorios hasta el archivo. Un nombre de camino relativo define el camino desde el subdirectorio actual.

Sistemas de archivos compartidos.

Una característica importante de un sistema multiusuario de tiempo compartido es la capacidad para compartir archivos, debido a que en forma simple y conveniente aumenta la productividad individual al permitir que programas y datos útiles en forma general los utilicen varios usuarios. La tarea del administrador se facilita ya que es posible que nuevos compiladores y utilerías estén disponibles a todos los usuarios en forma de archivos compartidos.

El compartir archivos no es crear alias de archivos; un archivo compartido tiene un solo nombre de ruta, mientras que un archivo con alias está identificado por dos o más nombres de ruta. El compartir archivos puede organizarse de forma individual o en grupo. En cualquiera de los dos casos, un archivo tiene un solo propietario y el propietario tiene control sobre la manera en que se permita si requiere compartir ese archivo. Compartir en forma individual implica que el propietario puede designar usuarios específicos que tendrán acceso a un archivo, esto tiende a generar complicadas estructuras de directorio, y por lo general los actuales sistemas limitan la capacidad de compartir archivos. Compartir en grupo significa que el propietario puede ampliar el acceso sólo a ciertos grupos de usuarios.

Protección de archivos:

Cuando la información se guarda en una computadora, es importante su protección del daño físico (seguridad) y del acceso impropio (protección).

La seguridad es, generalmente proporcionada por la generación de copias duplicadas de archivos. Muchos de los sistemas operativos copian automáticamente archivos de disco a cinta magnética en intervalos regulares de tiempo para mantener una copia, debido a que un archivo del sistema puede ser accidentalmente destruido. Los archivos del sistema pueden ser dañados por problemas de hardware o software, tales como errores en la lectura y escritura, fallo en la corriente eléctrica, rotura de la cabeza, o problemas externos como suciedad, temperatura o vandalismo.

Seguridad Externa.

Dentro de la seguridad externa se considera la seguridad física y operacional. La primera incluye la protección contra desastre y personas ajenas al sistema de cómputo; los mecanismos de detección son importantes para ésta. Los detectores de humo y los sensores de calor pueden emitir una oportuna señal de alarma en caso de incendio.

Respecto al problema de personas extrañas algunos detectores pueden determinar si un intruso ha entrado al lugar donde se encuentra el equipo o si ha tenido acceso al sistema sin tener derecho a éste. El costo de la protección contra desastres puede ser elevado por lo cual muchas veces ésta es tratada a la ligera, esto depende en gran medida de las pérdidas estimadas que se tendrían en un caso de desastre.

La seguridad operacional consiste de varias políticas y procedimientosimplementados por el encargado de la instalación del equipo de cómputo.

Políticas:

- La autorización determina los accesos permitidos para cada archivo.
- La clasificación divide el problema en subproblemas; los datos del sistema y los usuarios se dividen en clases, a las cuales se conceden diferentes derechos de acceso. Esto debido a confidencialidad de cierta información del sistema.
- La división de responsabilidades es una consecuencia de la clarificación, con ella se pueden dar distintos conjuntos de responsabilidades con las cuales los usuarios desarrollan su trabajo sin la necesidad de conocer todo el sistema.

Procedimientos:

Protección con password: Este procedimiento consiste en asignar un código que sea una cadena larga de varios caracteres. Esta cadena va estar grabada en la memoria. Cuando el usuario quiera tener acceso a la información tendrá que digitar su password que se comparará con el almacenado en memoria. Si el password digitado coincide con el de la memoria, el usuario podra comenzar a trabajar con el sistema.

El password tiene algunas desventajas; las contraseñas largas son difíciles de recordar, por lo que muchas veces los usuarios tienden a anotarlas, facilitando que alguien se entere de la contraseña, por tal motivo, los diseñadores deben elegir un esquema que use contraseñas lo suficientemente largas como para hacer que los ensayos repetidos resulten inútiles, pero lo suficientemente cortas como para que sean recordadas.

Vigilancia:

La vigilancia de un sistema se realiza con procedimientos de monitoreo o auditoría además de permitir el acceso al mismo sólo a usuarios autorizados. Los auditores son llamados periódicamente

Un registro de auditoría es un control permanente de acontecimientos importantes sucedidos en el sistema de cómputo. Este registro se realiza de forma automatica cada vez que tiene lugar tal evento y se almacena en un área altamente protegida del sistema. Este registro es un importante mecanismo de detección aunque no garantiza una buena seguridad por lo que es necesario revisar frecuente y cuidadosamente el registro. Tales revisiones deben hacerse de dos formas: periódicas y al azar. Las primeras prestan atención regular a los sistemas, mientras las segundas ayudan a detectar usuarios no autorizados y/o acciones fraudulentas. Controles de acceso: La seguridad interna consiste en controlar el acceso a los datos almacenados. Los derechos de acceso definen el acceso que tienen los usuarios a los objetos dentro del sistema. Los objetos son entidades que contienen información. Pueden ser objetos concretos como discos, cintas, procesadores o palabras de almacenamiento, u objetos abstractos que corresponden a las estructuras de datos o procesos.

Los objetos están protegidos contra usuarios no autorizados. Los derechos de acceso más comunes son de lectura, escritura y de ejecución; una forma directa de implementarlos es mediante una matriz de control de acceso. Esta es una matriz bidimensional en la cual los distintos sujetos son listados en

las filas y los diferentes objetos a los cuales se solicita acceso se relacionan en las columnas. Cada ranura de la matriz contiene los derechos de acceso que ese usuario tiene a un objeto.

2.3.2 Manejador de Trabajos.

Se puede considerar al manejador de trabajos como un supervisor global que asigna los recursos de sistemas a determinados trabajos

Este debe llevar control de los trabajos, invocar las políticas para decidir los trabajos que reciben recursos, asignar y desasignar éstos. Por ejemplo en un sistema de tiempo compartido la política de planeación de trabajos puede consistir en admitir los treinta primeros que se registren. Para este caso un algoritmo simple de prioridad de dos niveles permite que un usuario con mayor prioridad force la salida de un usuario de menor prioridad. El usuario a desplazar se escoge en base al tiempo de procesador que se ha utilizado, es decir el que ha usado el mayor tiempo de procesador es desplazado.

En un sistema por lotes las políticas son distintas; en éste no sólo se toma en cuenta la hora a la que llega un trabajo, sino también la prioridad, las necesidades de memoria de dispositivos, de procesador, y el equilibrio del sistema.

Se puede considerar al manejador de trabajos como un supervisor global que asigna los recursos de sistemas a determinados trabajos.

Funciones:

1. Llevar control del estado detodos los trabajos. Debe llevar un registro de los trabajos que tratan de lograr un servicio (estado de retención) y el estado de todos los trabajos a los que se está sirviendo (listos, en corrida o estado bloqueado).
2. Elegir la política según la cual los trabajos entran al sistema es decir pasan del estado de retención al listo. Esta decisión puede basarse en características tales como prioridad, recursos solicitados o equilibrio del sistema.
3. Asignar los recursos necesarios para el trabajo planeado por uso de memoria, dispositivo, y administración del procesador.
4. Desasignar los recursos al terminar.

Políticas:

El manejador de trabajos debe escoger entre los trabajos en retención aquellos que deben prepararse para correr. Se puede almacenar primero todos los trabajos entregados en un dispositivo de almacenaje secundario de donde el planeador de trabajos puede examinar a todos y luego poder escoger los trabajos a los que se entreguen los recursos del sistema y por consecuencia correan.
Se debe considerar lo siguiente:

El concepto fundamental del manejador de trabajos es que hay mas trabajos que desean correr de los que pueden satisfacerse eficientemente mediante los recursos del sistema. La planeación se considera un asunto de políticas debido a que generalmente sus metas son subjetivas.

Las consideraciones típicas con que se debe trabajar al determinar una política de planeacion de trabajos son:

1. Disponibilidad de recursos limitados especiales por ejemplo cintas.

   a) Si otorgan preferencias alguien puede hacer trampas por ejemplo si siempre corre primero el trabajo que solicite un graficador, entonces algunos usuarios siempre solicitarán uno.
   b) Si no da preferencia, algunos usuarios sufren de demora adicional por ejemplo, deben esperar a que se monte una cinta o se ajuste el graficador.
2. Costo: mayores tarifas por servicios más rápidos.

3. Compromisos del sistema: tiempo y memoria del procesador, entre más desea, más debe esperar.
4. Equilibrio del sistema: mezcla de trabajos de E/S intensa y CPU intensa.
5. Servicio garantizado: establecer un límite específico de espera por ejemplo una hora o un límite general (dentro las 24 horas siguientes).

Una vez que el manejador de trabajos ha seleccionado una colección de trabajos por correr, este intenta manejar la planeación microscópica (asignación dinámica del procesador a los procesos).

2.4 Manejo de Entradas y Salidas

Evolución de las operaciones de E/S.

Los dispositivos, como las lectoras y perforadoras de tarjetas e impresoras tenían dos problemas que impedían su utilización efectiva. En primer lugar, si un trabajo trataba de generar solicitudes más rápidamente que la razón de rendimiento del dispositivo, el trabajo debía de esperar demasiado tiempo. Por otra parte, si el trabajo generaba solicitudes a una razón mucho más baja, el dispositivo permanecía ocioso gran parte del tiempo quedando parcialmente desocupado ello quedaba parcialmente desutilizado. En segundo lugar, estos dispositivos se dedicaban a un solo trabajo a la vez.

En un principio se utilizaron tres pasos para gestionar la información. En el primer paso se utilizaba un procesador cuya única función era leer las tarjetas a velocidad máxima y registrar la información correspondiente en el dispositivo de almacenaje de acceso directo. En el paso dos el DASD, que contenía la entrada registrada por el procesador uno, se mueve al procesador principal de procesamiento. Finalmente en el paso tres, el DASD de salida se mueve a un tercer procesador que lee la información a alta velocidad y la imprime. Al trabajo realizado por los procesadores uno y tres (al no realizar cálculos sino transferencia de información de un dispositivo periférico a otro) se le llamó procesamiento periférico fuera de línea, esto debido a que se realizaba en forma independiente del procesador principal.

La técnica de procesamiento fuera de línea resolvió los problemas planteados anteriormente, aunque introdujo nuevos problemas tales como la intervención humana, el tiempo de retorno y la planeación. Ya que necesitaban operadores humanos para mover los dispositivos de almacenaje de acceso directo de entrada del procesador periférico de entrada hasta el procesador principal y desde este al de salida.

Si se deseaban ejecutar dos trabajos de alta prioridad que se encontraban en lotes separados (DASD separados) se tenía que esperar hasta que se terminara de procesar completamente el otro lote. El problema de mover físicamente los DASD de E/S, fue solucionado haciendo que éstos estuviesen conectados físicamente, tanto al procesador periférico como al principal, con lo que se eliminaba la necesidad del manejo humano. Un sistema físico, acoplado directamente, podía utilizar un solo DASD compartido por la entrada y salida o varios DASD compartidos. Es importante coordinar cuidadosamente el uso de un DASD compartido para que, tanto el procesador periférico como el principal puedan utilizar el dispositivo de almacenaje de acceso directo al mismo tiempo.

Descripción general:

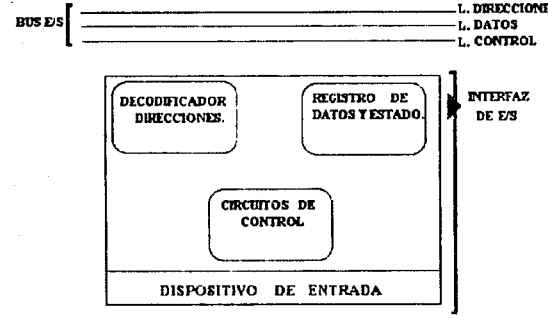
La conexión de un dispositivo de entrada y salida a un sistema necesita generalmente un circuito de interfaz, que puede consistir desde unos pocos registros o puertas lógicas hasta una o más placas lógicas. Cualquier instrumento que pueda producir una señal eléctrica o por pulsos, puede ser un dispositivo de entrada. De igual forma todo instrumento controlable por una señal eléctrica puede ser uno de salida.

La mayoría de las técnicas de entrada y salida e interfaces se han realizado en forma de circuitos integrados LSI(Large-Scale Integration) que se utilizan directamente para el control del dispositivo, estos controladores se conectan a los integrados de E/S estándar o, a veces a los buses estándares. Un controlador de discos emite órdenes de avance del mecánico mediante un número

concreto de pasos. Para realizar una transferencia de datos, la interfaz de E/S debe hacer posible lo siguiente:

1. Decodificación de la dirección que envía el CPU para seleccionar el periférico, esto implica que debe disponer de un decodificador de direcciones.
2. Captación o envío de los datos. El CPU manda o recibe los datos emitidos por los periféricos. Esto implica que debe de tener un registro de datos.
3. Sincronización de la transmisión. Deben de haber mecanismos que permitan la sincronización de puesta y recogida de datos. El registro de estados proporciona el estado del periférico en cada momento.

Estructura básica de un interfaz (figura 2.10)



E/S a nivel lógico y de dispositivos:

En un sistema operativo se utilizan discos para almacenar archivos, páginas de memoria virtual y/o imágenes de proceso para intercambio de segmentos. El diseño del hardware determina que el tamaño de página en memoria virtual tenga una relación simple con el tamaño de bloque del disco, y en un sistema simple de intercambios determina que la unidad de intercambio corresponda con un número entero de bloques de disco.

Por otra parte los archivos del usuario y archivos del sistema operativo son de tamaño variable y de varias clases. Estas dos necesidades en el disco se resuelven proporcionando una capa extra de software y se distinguen dos niveles de actividad de E/S, E/S lógica y E/S física.

1. E/S física es aquella entrada-salida en la cual la unidad de transferencia coincide con la unidad física adecuada al dispositivo, por ejemplo, las pistas de un disco.
2. E/S lógica es aquella entrada-salida en donde el programador selecciona el dispositivo que considera es más conveniente para cumplir sus objetivos, por ejemplo un dispositivo físico considera a un disco no como pistas de longitud fija sino como archivo de longitud variable. Las rutinas de E/S lógicas mapean archivos de longitud arbitraria sobre secuencias de bloques de tamaño físico, y llaman a las rutinas de E/S físicas para transferir información (bloque por bloque). El objetivo del sistema de E/S a nivel dispositivo es considerar todas las características dependientes del dispositivo de una clase específica de dispositivos periféricos, presentando así una interfaz uniforme a los niveles superiores del sistema operativo.

El manejador de dispositivo es un programa que corre como proceso,sincronizado al dispositivo por la señal (o señales) de interrupción y al proceso "usuario"por medio de un semáforo. Debido a que un manejador de dispositivo puede controlar a un grupo de dispositivos no existe una correspondencia uno a uno entre dispositivos y manejadores, sin embargo, cada dispositivo tiene un solo bloque de control de dispositivo (DCB: device control block) asociado con el. Con el fin de separar los detalles dejna transferencia especifica de E/S de las propiedades genéricas de un dispositivo, se utiliza un bloque de transferencia de E/S (IOTRB: I/O transfer request block) para contener los detalles de una transferencia.

El manejador del dispositivo se encuentra entre el proceso que efectúa la solicitud y el dispositivo periférico y se sincronizan utilizando dos semáforos: el de dispositivo y el de solicitud. El primero es la señal de interrupción que sincroniza al dispositivo y forma parte del DCB; el semáforo de solicitud proporciona la sincronización con el proceso usuario, éste pertenece al proceso que efectúa la llamada y el acceso se realiza a través de un apuntador contenido en el IOTRB.

Organización de prioridades.

La organización de prioridades cuando varios dispositivos solicitan simultáneamente acceso al bus, es un problema típico de resolución de prioridades para acceso a un recurso común. El recurso en común es el bus y hay una serie de dispositivos que pueden pedir acceso al bus para establecer comunicación.

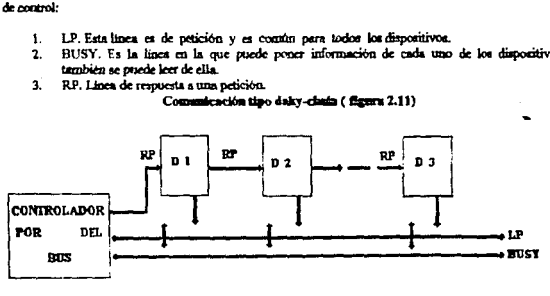
No se puede preveer cuando se van a producir las peticiones; por tanto, varios dispositivos pueden pedir simultáneamente acceso al bus. En este proceso la simultaniedad se refiere a un intervalo. Los métodos para resolver el conflicto de simultaneidad son:

Daisy-Chain (Cadena).

Se basa en una cadena que recorre todos los dispositivos peticionarios. El controlador del bus puede ser el CPU u otro dispositivo con la capacidad de controlador. En este caso hay tres líneas de control:

1. LP. Esta línea es de petición y es común para todos los dispositivos.
2. BUSY. Es la línea en la que puede poner información de cada uno de los dispositivos y también se puede leer de ella.
3. RP. Línea de respuesta a una petición.

Comunicación tipo daisy-chain (figura 2.11)



Sobre la línea de petición se pone un 1 como indicativo de petición cualquiera de los dispositivos. Cuando el bus quede libre se genera una señal de comprobación por la línea RP, que va primero al dispositivo 1, luego al 2 y así sucesivamente. La señal por la RP pasa de un dispositivo a otro, siempre que dicho dispositivo no haya hecho petición. Si el dispositivo 1 ha pedido acceso al bus, corta la señal de la RP para que no llegue a los demás. Cuando el primer dispositivo que haya pedido servicio reciba un 1 por RP, pone un 1 en la línea de BUSY y sólo lo hace el, este 1 es indicativo del comienzo de la transmisión.

Desventajas:

a) El orden de prioridades es fijo, es decir lo fija el orden en que están encadenados los dispositivos. Si se requiere cambiar el orden de servicio, se tienen que encadenar los dispositivos nuevamente de forma distinta.
b) Es un sistema muy poco tolerante a fallos del hardware. Si el dispositivo 1 falla y se pone fuera de servicio, el resto de los dispositivos tampoco podrán trabajar.
c) El dispositivo que haya hecho la solicitud tarda un tiempo largo en tener el control del bus, y este tiempo depende del número de unidades conectadas, ya que la señal por la RP se ha de propagar hasta el dispositivo solicitante de atención.

- Consultas sucesivas (Polling).

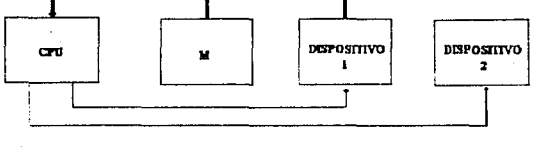
En este método los dispositivos de E/S están conectados de manera normal a los buses de datos y direcciones del sistema a través de los integrados de interfaz. La finalidad de toda técnica de gestión es obtener un procedimiento que determine el siguiente dispositivo de E/S que requiere servicio.

Ventajas:

1. Requiere un mínimo de hardware y no se necesitan líneas especiales.
2. La E/S se produce cuando la solicita el programa de ejecución.
3. La posibilidad de cambiar el orden de consulta de los dispositivos.

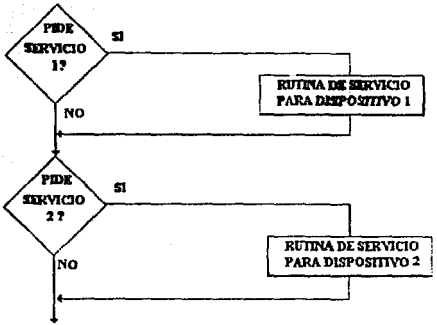
La técnica de consultas sucesivas es de tipo síncrono, ya que no interrumpe el programa en ejecución es decir la operación de E/S se produce en el mismo instante que lo necesita el programa. Con el polling el procesador se encarga de preguntar a cada dispositivo conectado al sistema si necesita servicio. El dispositivo contestara con un SI o con un NO. Si se recibe una contestación negativa, el procesador pregunta si necesita servicio el siguiente dispositivo.

Comunicación tipo polling (figura 2.12)



Esta consulta se realiza comprobando el estado de un bit especial llamado indicador o flag, que se encuentra en cada registro de estado del dispositivo o en su interfaz. El programa para realizar un algoritmo de consulta utilizado se denomina bucle de consulta, y se ejecuta según la secuencia mostrada en el siguiente algoritmo:

Algoritmo tipo polling (figura 2.13)



Antes de transmitir información a dispositivo, el CPU comprueba el bit de estado en el integrado interfaz de E/S para saber si el dispositivo está o no listo para aceptar datos. De forma similar, antes de leer una palabra de información de un registro de E/S, el CPU comprueba un bit de estado nuevamente para saber si en efecto, el registro está lleno.

Desventajas:

1. Cada vez que se encuentre en un bucle de consulta, hay que comprobar todos los dispositivos. Lo cual es un proceso largo y lento.
2. El procesador esta detenido durante la operación de E/S.

Interrupciones de *Entrada/Salida*.

Con las interrupciones, el procesador no está parado mientras se produce una operación de E/S. La ventaja principal de las interrupciones es la rapidez de respuesta. Las interrupciones se necesitan normalmente en sistemas que operan en un tiempo mínimo de respuesta a solicitudes externas. La detección del dispositivo interruptor depende del sistema empleado para introducir las interrupciones. Son iniciadas por el hardware de E/S, éste envía señales al CPU indicando que el estado de un canal o dispositivo ha cambiado.

Interrupciones de un nivel o línea.

Este tipo de interrupciones es un mecanismo de polling (preguntas sucesivas) y también son conocidas como interrupciones a nivel software. Se va leyendo el registro de control y estado de cada uno de los periféricos y aquel en que se encuentre su bit igual a 1 sera el que produjo la interrupción.

Debido a lo anterior el programa de identificación del periférico es un conjunto de preguntas sobre el bit de interrupción de cada uno de los periféricos. Cuando encuentre el bit con valor 1, lo

atenderá y cuando la interrupción esté servida el periférico borrará su flag es decir pondrá un 0 en el bit.

Puede ser que más de un periférico haya producido una interrupción en la misma línea, entonces el primero en atender es el primero que haya hecho la solicitud. La prioridad la fija el orden de las preguntas en el polling. La gestión de prioridades en el polling puede ser:

- Sistema fijo de prioridades.
- Sistema rotante de prioridades.

Si dispone una cadena de periféricos, al producirse una interrupción, el periférico prioritario es el que está a continuación del servido en la cadena. Por ejemplo, si se sirve al periférico 5, la vez siguiente el primero por preguntar es por el periférico 6. En cada momento la prioridad más alta la obtiene el que tiene el número de orden siguiente al último servidor.

Interrupciones de nivel simple y vectorado :

A estas interrupciones también se les llama interrupciones a nivel hardware o sistema de interrupciones vectorado. Aquí cada periférico proporciona al CPU su identificación, que consiste en dar la dirección de comienzo de la rutina específica de tratamiento.

Cuando la dirección del salto al tratamiento de la interrupción producida esté disponible por hardware se le denomina interrupción vectorada.

Cada controlador de periférico a través de su registro de estado pide interrupción; una vez que el CPU ha recibido la señal de interrupción, genera una señal de respuesta, esta señal llega al primer dispositivo (periférico) y, si éste no ha interrumpido, la deja pasar al segundo y así sucesivamente. Si el periférico se ha interrumpido, pone mediante otro registro su identificación en el bus de datos. El CPU leera el bus de datos en el que se encuentre la clave o dirección del periférico que se ha interrumpido, con lo cual se obtiene la dirección específica del programa de tratamiento del periférico que ha interrumpido; mientras que el anterior tipo de interrupciones se bifurcaba al programa de tratamiento que es común para todos los periféricos.

Interrupciones de nivel múltiple y vectorado.

La rutina de servicio de tratamiento de una interrupción de un nivel puede ser interrumpida por una petición de otro nivel más prioritario. Muchos procesadores tienen más de una línea de interrupción, entre las que se distribuyen los diferentes periféricos. Se agrupan en la misma línea los periféricos de características similares. Al distribuir los periféricos en más líneas, el tiempo de identificación es menor.

Dentro del procesador, existe un codificador de líneas de interrupción. Este codificador da como salida el código de la línea activada de mayor prioridad y una señal de interrupción pendiente en caso de que haya líneas de más alta prioridad.

Canales de Entrada/Salida.

A estos canales también se le conoce como procesadores de E/S (P I/O). El canal se comunica con los periféricos mediante el mismo camino que el procesador central, y se comunica con los mismos bloques de memoria que el procesador central, pero por distinto camino; por tanto se necesita memoria con dos entradas y con lógicas diferentes.(memoria multipuerta).

Características:

1. Las características de los procesadores (pueden ejecutar programas), pero son mucho menos potentes que el procesador central.

2. Son procesadores dedicados a operaciones de E/S. Tienen su propio repertorio de instrucciones que ejecutan autonáticamente.
3. Suelen ser procesadores esclavos, es decir, están supervisados por el procesador central.

Funciones:

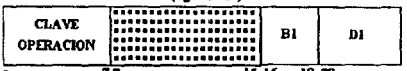
1. Debe de ser capaz de seleccionar un periférico particular para realizar una operación de E/S.
2. Debe enviar comandos (funciones de control) al periférico.
3. Definir áreas o registros en memoria o periféricos para la transferencia.
4. Definir la acción a seguir cuando se acabe la transferencia completa.
5. Detectar el estado del periférico.

El procesador central solo le indica al canal de E/S que programa va a realizar una operación de E/S. Estos canales (P E/S) sólo se implementan en sistemas medios y grandes.

Instrucciones de E/S.

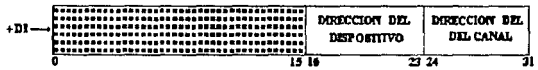
Todas las instrucciones de E/S del CPU tienen el siguiente formato:

(figura 2.14)



Los números de canal y dispositivo se especifican por la suma de los contenidos del registro B1 y el campo D1. Los bits 16-23 de la suma contienen la dirección del canal, en tanto que los bits 24-31 contienen el dispositivo en el canal.

(figura 2.15)



Todo el grupo de instrucciones de E/S se puede dividir en dos subgrupos:

1. CCW: Son las instrucciones que ejecuta el P E/S y son:

   a) Leer/escribir.
   b) De control de programas: bifurcación.
   c) De control de periféricos. Instrucciones de control sobre los diferentes periféricos.

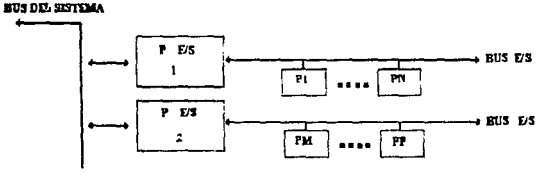
2. Instrucciones de E/S que ejecuta el procesador central y son:

   a) START E/S: Inicializa la ejecución de un programa de E/S en un P E/S determinado.
   b) HALT E/S: Finaliza la ejecución de un programa que esté en un canal.
   c) TEST E/S: Para ver cuál es el estado de los P E/S.

Estructuras de interconexión entre P E/S y periféricos.

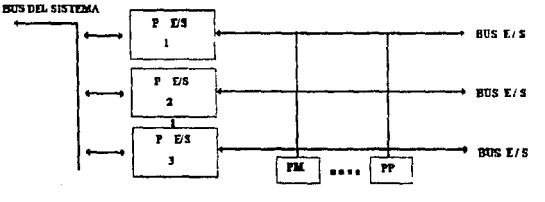
1. Conexión en serie: Se tiene un bus E/S por cada canal de E/S (P E/S), en cada uno de los cuales están colgados determinados periféricos con sus correspondientes controladores. Si se quiere comunicar con un determinado periférico, hay que hacerlo a través del P E/S al que pertenece dicho periférico. Esta estructura es la más barata, pero no es muy flexible ya que tiene el inconveniente de que no se puede comunicar con dos periféricos que están colgados del mismo canal de entrada P E/S, aunque el resto de los P E/S estén libres. La comunicación se realizara dentro de una misma línea con el periférico que tenga mayor prioridad.

Conexión en serie de los periféricos al P E/S (figura 2.15)



2. Matriz de conexiones dinámicas: Esta conexión permite que todos los P E/S estén ocupados mientras haya peticiones y procesadores libres; pero esta estructura es más cara que la anterior. Permite que cada procesador de P E/S se conecte con cualquier periférico.
Cada punto de cruce tiene su propia lógica y se puede comprobar si cada punto de cruce está activado o no. De esta forma se pueden tener tantas transferencias en ON como procesadores de E/S tengan. Esta conexión tiene un problema: se adapta muy mal al crecimiento por ejemplo si se requiere añadir un periférico o un procesador de E/S se tiene que rediseñar la lógica de la matriz, ya que cambiaría la dimensión.

Conexión en red de los periféricos al P E/S (figura 2.17)



Hardware necesario:

Un procesador (canal de E/S) necesita tener básicamente:

- Un buffer de datos.
- Una unidad de control que sea capaz de buscar en memoria, de decodificar e interpretar las instrucciones y de realizar funciones de control sobre los periféricos.

Tipos de canales de E/S.

Canal Multipléxor.

Un canal multipléxor en bloque es una solución intermedia que permite que programas de canal múltiple de dispositivos de alta velocidad estén activos en el mismo canal de E/S. El multipléxor en bloque realiza una instrucción de canal para un dispositivo, y luego automáticamente cambia para realizar una instrucción para otro dispositivo y así sucesivamente.

El tiempo de proceso del canal (procesador) se reparte entre todos los periféricos conectados a el. Cada subcanal hace operaciones de E/S con un periférico.
Para realizar las operaciones de E/S cada subcanal debe tener el siguiente conjunto de registros:

- Buffer de datos de información transmitida, donde se guarda el carácter hasta que es transmitido a la memoria.
- Registro de estado; donde se refleja el estado en que ha quedado la transferencia.
- Registro de dirección de memoria (fija). Es un puntero a la memoria del procesador central, donde están guardados los parámetros de la transferencia que se va a realizar.

Cada instante que el canal (procesador) de E/S atiende a un periférico, el subcanal es multipléxor porque reparte su tiempo entre los periféricos que están conectados a el. El procesador de E/S que además de tener un canal multipléxor tiene una lógica buffer que permite convertir información que viene de byte en byte (8 bits) a información de 16 bits de longitud. Los dispositivos a los cuales atiende este tipo de canal son de velocidad media o baja.

Canal selector.

Este tipo de canal de E/S se utilizan para conectar periféricos que trabajan a alta velocidad, como cintas y discos magnéticos. Un canal selector selecciona sólo un dispositivo a la vez para darle servicio.

En este caso el procesador no tiene tiempo de atender a más de un periférico a la vez; por eso, cuando inicia una transferencia en un periférico, no atiende a ningún otro hasta que termine su operación. En vez de almacenar en memoria central los parámetros de la transferencia, como el multiplexado, se almacenan en los registros internos del mismo procesador de E/S; por lo tanto solo necesita acceder a memoria para depositar datos.

El canal selector tiene:

- Un registro de direcciones de memoria de acceso aleatorio que le indica de dónde se deben de leer o escribir los datos.
- Un contador indicativo del número de palabras a transmitir llega a valer 0 cuando termina la transmisión. En cada transferencia se actualizan los contenidos de estos registros.
- Un registro DIR.PERIFE, que contiene la identidad del periférico con el que se va a hacer la operación.
- Hay un registro de ensamblado (operación de unir bytes para formar 16 bits) que se usa para entrada de datos y dispone de un chequeo de errores. Sólo permite el paso de datos correctos.
- Un registro de desensamblado (operación que descompone una transmisión en bytes) que se utiliza en la salida y proporciona información redundante para errores.

Canal multiplexor por bloques.

Es una mezcla del canal multiplexor y el canal selector, aunque se parece a un selector. Se utiliza para conectar dispositivos de alta transferencia.

El canal selector es rápido, siempre tiene tiempos muertos, mientras busca la pista del disco y posiciona la cabeza sobre el registro seleccionado. En este tiempo muerto el procesador selector no hace operaciones de E/S.

Desde que se da la orden al disco de que mande determinada cantidad de información hasta que realmente se transmite, pasa un cierto tiempo. Durante el tiempo de acceso se aprovecha para hacer transferencias de otros dispositivos que estuvieran ya preparados.

Comunicaciones entre el CPU y el CANAL.

El propósito de tener un canal es liberar el CPU de tener que controlar las operaciones detalladas de E/S. EL CPU y el canal están en una relación de maestro/esclavo, lo que significa que el CPU indica al canal cuando debe comenzar y le ordena detenerse o cambiar lo que hace. Por otra parte, por lo general el canal no puede iniciar ninguna operación a menos que así lo indique el CPU.

Existen dos tipos de comunicaciones entre el CPU y el canal:

a) Instrucciones de E/S de CPU al canal iniciadas por éste.
b) Interrupción de canal al CPU iniciada por el canal.

Administrador de dispositivos.

El manejo de E/S esté controlado totalmente por la administración de recursos la cual segun sus funciones se divide en tres partes:

1) Controlador de tráfico de E/S: Lleva el control del estado de todos los dispositivos, unidades de control y canales.
2) Planeador de E/S. Implementa los algoritmos de política empleados para asignar el acceso al canal, unidad de control y dispositivo correspondiente a solicitudes de E/S de los trabajos.
3) Manejadores de dispositivos de E/S. Efectúan las operaciones dinámicas físicas, una vez que el planeador de E/S ha tomado la decisión, construyendo el programa de canal emitiendo la instrucción que arranca la operación de E/S, y procesando las interrupciones de dispositivo.

En tanto que el planeador de E/S se preocupa principalmente por las políticas de asignación, el controlador de tráfico de E/S se interesa principalmente en el hardware es decir si puede o no asignarse el dispositivo. El controlador de tráfico mantiene también toda la información de estado.

El controlador de tráfico de E/S se complica debido a las interdependencias introducidas por la escasez de canales y unidades de control. Este trata de responder tres preguntas claves:

1. ¿ Hay alguna ruta disponible para atender una solicitud de E/S?
2. ¿ Hay más de una ruta disponible?
3. Si actualmente no se dispone de una ruta. ¿ Cuándo habrá una libre?

Para responder esto el controlador de tráfico mantiene una base de datos que refleja el estado y las conexiones, lo que se logra mediante bloques de control de unidad (UCB), bloques de control de unidad de control (CUCB), y bloques de control de canal (CCB).

Si hay más solicitudes de E/S pendientes de rutas disponibles, es necesario elegir cuales satisfacer primero. Para esto se incorporan muchas políticas diferentes del planeador de E/S. Por ejemplo, si el planeador de procesos asigna alta prioridad a un proceso, también es razonable asignarle alta prioridad a sus solicitudes de E/S lo que ayudaría a concluir el trabajo más rápido.

Una vez que el planeador de E/S ha *determinado* las ordenes relativas de las solicitudes de E/S, el controlador de tráfico de E/S debe determinar cuál puede satisfacerse en caso posible.

Además de preparar las palabras de comando de canal, manejar las condiciones de error y procesar las interrupciones de E/S, los manejadores de dispositivos de E/S proporcionan algoritmos de planeación detallada que dependen de las peculiaridades del tipo de dispositivo. Generalmente existe un algoritmo distinto de manejador de dispositivo para cada tipo de dispositivo de E/S.

Almacenamiento físico en disco.

La superficie de un disco está formateada físicamente en pistas direccionables, las cuales pueden subdividirse en sectores direccionables. En un disco de múltiples superficies las cabezas de lectura y escritura se mueven juntas, de manera que con dichas cabezas posicionadas en una pista específica es posible leer la pista correspondiente en todas las superficies. A este conjunto de pistas que puede leerse sin movimiento de las cabezas se le denomina cilindro, y el direccionamiento en disco se realiza de manera que las direcciones consecutivas vayan a través de las pistas de un cilindro y después a las del siguiente. Externamente, una dirección en disco es de la siguiente forma: número de cilindro/número de pista.

2.5 INTERFAZ DEL USUARIO.

Para todo sistema operativo es necesario que se comunique con los usuarios de alguna manera. Existen dos formas de lograrlo mediante llamadas al sistema las cuales son mensajes al nucleo que permiten que los programas manipulen archivos, dispositivos y procesos, las cuales son:

1) Lenguaje de control que está constituido por un intérprete de comandos, el cual toma los comandos provenientes del teclado. Y el usuario lo tiene que aprender.
2) Por medio de menús que el sistema despliega en la pantalla, para que el usuario seleccione la operación que desea realizar.

Ambas formas tienen ventajas y desventajas, aunque son más los sistemas operativos que manejan el concepto de lenguaje de control que el de menús.

En los sistemas operativos existe una distinción importante entre aquellos sistemas que permiten acceso abierto a las llamadas del sistema y aquellos que lo restringen. En general es posible que a los programas normales se les niegue cualquier acceso directo a las llamadas del sistema, los sistemas de este tipo permiten que un programa genere una corriente de texto que interpretara como si se hubiera escrito en el teclado. La consecuencia importante de limitar el acceso a las llamadas del sistema es que esta limitación hace que el intérprete del lenguaje de comandos sea una parte indisoluble del sistema operativo.

Por otra parte los sistemas que permiten el acceso libre a las llamadas del sistema sólo están sujetos a tener permisos necesarios para leer/ejecutar los archivos deseados. En el acceso libre el intérprete de lenguaje de comandos es tan sólo otro programa, lo cual permite ofrecer varios comandos ya que en realidad no son muy especiales. De esta forma es posible proporcionar una interfaz simple de menú para los usuarios novatos.

El lenguaje de control proporciona interfaz denominada interfaz interactiva la cual lee lineas de comandos a partir del teclado y efectúa las acciones especificadas. Un comando consiste en un

nombre de comando seguido quizás por uno o más argumentos. El intérprete del lenguaje de comandos obedece directamente a algunos comandos simples, pero la mayor parte de los comandos requieren de cargar un programa a partir del disco. En estos casos el programa de comando debe transferir el control de nuevo al intérprete del lenguaje de comando al terminar ya sea de forma normal o anormal.

En un sistema operativo multiusuario es necesario cierto tipo de protección para asegurarse que un usuario no pueda tener acceso ilícito a los archivos de otros usuarios, y el primer componente de la interfaz interactiva es la autenticidad del usuario que intente tener acceso al sistema. Por lo general, esto se hace solicitando una clave de acceso (password) que conoce el sistema y permite el acceso.

El lenguaje de comando define el ambiente en el cual tiene lugar el uso interactivo del sistema. Los primeros sistemas operativos tenían un conjunto fijo de comandos, interconstruidos o integrados al intérprete del lenguaje de comandos. En éstos se proporcionaba un mecanismo general de escape por medio de un programa RUN que provocaba que un programa en un archivo con nombre se corriera. De esta forma podía lograrse el acceso a subsistemas.

Los sistemas modernos utilizan la técnica de tomar el nombre del comando como nombre de un archivo ejecutable y obteniendo el mismo del disco. Esto proporciona un recurso general y ampliable por completo.

Resulta conveniente que los comandos tengan un formato uniforme y sigan convenciones uniformes acerca de los argumentos. En los sistemas multiusuario los comandos pueden tener cualquier número de argumentos.

2.5.1 Interfaz por lote.

El trabajo por lote está restringido a las instalaciones de procesamiento comercial de datos y a sistemas científicos grandes que trabajan con cifras y cálculos voluminosos; en ambos casos esto se realiza en grandes macrocomputadoras que operan bajo control de elaborados y complejos sistemas operativos.

En un sistema operativo basado en procesos el lector, escritor y supervisor de trabajos se vuelven procesos y el núcleo los corre en forma concurrente junto con los otros procesos del sistema.

Los recursos de la interfaz por lote son los de los sistemas de manejo por spool. En el caso más simple, el usuario presenta un trabajo en forma de conjuntos de tarjetas y estos trabajos se juntan para manejarlos por lote y se corren en secuencia. Por lo general un trabajo por lote se presenta al sistema operativo como una descripción de trabajo escrita en un lenguaje de control de trabajos especializado. Esta descripción del trabajo muestra los pasos del trabajo y su secuencia detalla los archivos que deban utilizarse en cada paso define las conexiones que se efectuaran entre estos archivos y los "canales" de entrada-salida vistos por los programas que incluye el trabajo. La descripción del trabajo incluye información de manejo cronológico y la especificación de límites acerca del empleo de recursos como por ejemplo el tiempo de CPU.

2.5.1.1 Asignación de recursos en sistemas por lote.

Un problema que surge en los sistemas por lote, es la asignación de recursos no compartibles.

En un ambiente interactivo puede indicársele al usuario si el recurso que desea se encuentra disponible y puede decidir por si mismo que acción realizar. En un sistema por lotes, por definición el usuario no tiene posibilidad de intervenir en la corrida del trabajo, porlo que el sistema operativodebe administrar los recursos.

La estrategia más simple consiste en asignar todos los recursos que necesite un trabajo antes de correrlo, de esta forma el sistema puede asegurar que el trabajo sera capaz de terminar la corrida con éxito. Sin embargo, esta estrategia puede provocar que recursos escasos permanezcan ociosos durante periodos largos. Por esta razón resulta conveniente adoptar una estrategia de asignación dinámica en la que los programas soliciten recursos cuando en realidad los requieran, de manera que aumente el uso de recursos en un ambiente de multiprogramación. En este caso el sistema operativo debe asociar una línea de espera de solicitudes para asignación de un dispositivo no compartible, de manera que los programas en competencia tengan un dispositivo en orden en el

momento de solicitud, o de acuerdo con algún margen de prioridad, según lo determine el administrador del sistema.

Si los programas tienen acceso a varios recursos no compartibles, existe el riesgo de que ocurra un punto muerto. Por ejemplo supóngase que dos programas P1 y P2 requieren los recursos R1 y R2 y que la secuencia es como sigue:

P1:.....solicita R1.....solicita R2.....solicita R1.....solicita R2
P2:.....solicita R2.....solicita R1.....solicita R2.....solicita R1

Si los dos programas están multiprogramados juntos existe la posibilidad de un punto muerto si P1 solicita R2 antes de que P2 lo haya liberado, y la solicitud de P2 de R1 se haga antes que P1 lo haya liberado. Si esto ocurre, ambos procesos se detendrán, cada uno en espera de que el otro haga algo, y salga del punto muerto.

2.5.2 Lenguaje de comando.

El lenguaje de comando es la interfaz principal entre el sistema, el usuario y los recursos que permite la comunicación entre los mismos. Históricamente, los lenguajes de comandos tanto para ambientes interactivos y como para ambientes por lotes se han desarrollado siguiendo distintas rutas.

A primera vista, la necesidad de lenguajes de comandos para trabajos interactivos parece ser trivial. Los comandos consisten en un nombre de comando (o verbo) seguido probablemente de argumentos, los cuales son obedecidos uno cada vez conforme los teclea el ususario. Las principales diferencias entre sistemas se encuentran en la variedad de comandos disponibles y en la forma que el sistema direcciona al usuario.

CAPITULO 3.

Estudio del Sistema Operativo UNIX

Una vez que el estudiante es capaz de reconocer los componentes del sistema operativo así como las funciones de los mismos, está preparado para poder aprender fácilmente cualquier Sistema Operativo. La selección de Unix como caso de estudio se debe a que no es un sistema operativo monolítico, como casi todos los demás, sino que está compuesto de un pequeño núcleo y un conjunto de rutinas y operadores, lo cual facilita identificar todos los componentes que el alumno estudio en la parte teorica del curso.

Unix fue diseñado para servir de entorno en las labores de diseño de producción de programas lo cual será una de las futuras actividades del egresado de MAC así que ésta es otra razón de su elección para este trabajo.

Aunque es un sistema operativo muy completo tiene ciertas desventajas entre ellas:

- Es relativamente caro en recursos.
- Requiere de un disco rigido, rapido y eficaz.
- Requiere de velocidad del procesador central.
- No todo es accesible en las computadoras personales debido a que UNIX no fué diseñado pensando en ellas.

El sistema Unix actualmente no es tan popular como otros Sistemas Operativos, pero dadas circunstancias en el mercado hacen indicar que su futuro es excelente.

3.1. Historia.

El sistema operativo UNIX fue desarrollado para comercializarse en los Laboratorios Bell a principios de la década de los setentas. Su codigo fuente está escrito en lenguaje C, lo cual lo hace un sistema muy portatil. Los usuarios pueden dirigir la salida de un programa a la entrada de otro, haciendo posible implementar programas grandes mediante el montaje de los programas existentes.

La filosofía de operación UNIX esta basada en el concepto de herramientas de software el cual requiere que las tareas computacionales se construyan paulatinamente, donde el sistema aporta un conjunto de operaciones primitivas, que el diseñador usa para crear operaciones básicas. Es decir, con un pequeño número de funciones elementales se pueden configurar programas y sistemas completos que cumplan una función especifica.

Los sistemas UNIX estándar son sistemas operativos multiprogramables de tiempo compartido, en un principio diseñados principalmente para computadoras PDP-11/34, 40, 45 y 70 de DEC.

La historia del sistema operativo UNIX se describe a continuación:

1965: Las empresas Bell Telephone Laboratories y General Electric Company intervienen en el proyecto MAC del Massachusets Institute Tecnology para desarrollar un nuevo sistema operativo denominado MULTICS, cuyo objetivo era ofrecer un sistema multiusuario de gran potencia de proceso, capacidad de almacenamiento y grandes facilidades para compartir datos entre procesos.

1969: Debido a los buenos resultados de MULTICS, la Bell Telephone Laboratories se retira del proyecto y desarrolla un sistema de tiempo compartido con paginación por demanda para uso interno de la empresa. El nuevo proyecto fue encabezado por Ken Thompson y Dennis Ritchie.

Este sistema constituyó la primera versión del UNIX, que sólo permitía la explotación en monoprogramación.

1971: El resultado del sistema anterior tuvo tanto éxito que la compañía puso a disposición de Thompson y Ritchie una computadora más potente (la PDP-11 de Digital). En la cual Thompson desarrollo el lenguaje de programación B inspirándose en el BCPL y en el FORTRAN y después Ritchie creó el lenguaje C, con el que consiguió la generación de código máquina, descripción de datos y estructuras de datos.

1973: Se reescribe en C la versión en *UNIX* desarrollada en ensamblador, la cual prácticamente es la que se ha *mantenido* hasta hoy. Aparece una versión de UNIX conocida como Programmer's Workbench (PWB).

1974: Se introduce el sistema operativo UNIX en las Universidades norteamericanas con fines educativos.

1977: Se construye la primera versión comercial del UNIX, conocida como versión 6, implantándose por primera vez en una computadora distinta de la PDP, que fue la INTERDATA 8/32.

1979: Aparece la versión 7 de UNIX para PDP y una versión para la computadora VAX de Digital (32 bits) conocida como 32V.

1981: Nace la primera versión de UNIX para computadoras personales con el nombre de XENIX.

1982: Para la distribución externa, los Laboratorios Bell desarrollan el UNIX System III, que es la versión original con pequeñas variantes. La Universidad de Berkeley desarrolla una variante del UNIX 32V para computadoras VAX con mejoras en cuanto a comandos y gestión de la memoria paginada, denominada 4.1 BSD.

1983: La empresa AT&T anuncia una nueva versión denominada UNIX System V, que es el sistema actual y que presenta importantes mejoras de rendimiento, comunicaciones, etc. 1984: La Universidad de Berkeley presenta la versión 4.2 BSD para computadoras VAX, que también se aplica en estaciones de trabajo SUN 2/3 de SUN MICROSYSTEMS.

Los sistemas UNIX estándar proporcionan un sistema de archivos jerárquico con protección total, volúmenes desmontables, independencia de dispositivos y características que facilitan la sencillez de la programación. Cualquier programa o grupo de programas puede ser ejecutado asincrónicamente de forma interactiva o subordinada sin cambio alguno.

Los sistemas UNIX no distinguen entre programas de usuario y programas del sistema, ni en capacidad ni uso excepto por las restricciones impuestas por la protección del archivo. El buffer de entrada/salida, las asignaciones de almacenamiento principal y de almacenamiento de disco son manejados automaticamente por el sistema y son invisibles al usuario.

Los sistemas operativos UNIX estándar se distribuyen con una serie de programas empaquetados que incluyen un editor de textos, un intérprete de lenguaje de comandos programable, varios compiladores para lenguajes populares, un ensamblador, un editor encadenador, depuradores formateadores de documentos (con posibilidades matemáticas), programas de procesamiento de textos, un dispositivo de clasificación, una capacidad de investigación de estado, capacidad de comunicación entre usuarios, programas administrativos y de mantenimiento, bibliotecas normales del sistema y rutinas del usuario, y un paquete de juegos.

El código fuente para todos los programas (excepto para el paquete de juegos) se suministra con el sistema.

3.2 Objetivos de UNIX.

1) Conservar la sencillez del mismo y apoyarse en tan solo una cantidad mínima de funciones. A los programas de usuario se les deja la tarea de preveer el uso de más procedimientos o funciones.
2) Generalidad, es decir, un solo método debe servir a diversos propósitos, éste se observa en los sistemas UNIX en varias áreas por ejemplo:

   - El sistema usa las mismas llamadas para leer o escribir archivos, dispositivos, y buffers de mensajes entre procesos.
   - Se aplican los mismos mecanismos de nomenclatura, formación de seudónimos, y protección de acceso a los archivos de datos, directorios y dispositivos.

3) Crear un ambiente en el cual las grandes tareas puedan ser cumplidas combinando pequeños programas existentes, en vez de desarrollar nuevos programas.

3.3 Características del Sistema Operativo UNIX:

- Es un sistema operativo multiusuario, con capacidad de simular multiprocesamiento y proceso no interactivo.
- Está escrito en lenguaje C.
- Dispone de un lenguaje de control programable, llamado Shell.
- Ofrece facilidades para la creación de programas, sistemas y un ambiente propio para las tareas de diseño de software.
- Emplea manejo dinámico de memoria ya sea por intercambio o paginación de memoria.
- Tiene capacidad de interconexión de procesos.
- Emplea un sistema jerárquico de archivos, con facilidades de protección de archivos, cuentas y procesos.
- Usa manejo consistente de archivos de diversos tipos.
- Tiene facilidades para redireccionamiento de entradas/salidas.
- Incluye más de un centenar de subsistemas, y varios lenguajes de programación.
- Garantiza un alto grado de portabilidad.

3.4 Componentes y Funciones del Sistema Operativo UNIX:

3.4.1 Núcleo

El sistema operativo UNIX se basa en un núcleo conocido como kernel que reside permanentemente en la memoria y atiende todas las llamadas del sistema, administra el acceso a los archivos y el inicio o suspensión de las tareas de los usuarios. El núcleo es un programa de aproximadamente 10000 líneas, escrito casi en su totalidad en lenguaje C, debido a que parte del manejo de interrupciones está escrito en el lenguaje ensamblador del procesador en el que opera.

Funciones:

- Permitir la existencia de un ambiente en el que sea posible atender a varios usuarios y múltiples tareas en forma concurrente, repartiendo al procesador entre todos ellos, intentando mantener en grado óptimo la atención individual.
- Operar como asignador de recursos para cualquier proceso que necesite hacer uso de las facilidades de los mismos.
- Creación de procesos, asignacion de tiempos de atención y sincronizacion.
- Asignación de la atención del procesador a los procesos que lo requieren.
- Administración de espacio en el sistema de archivos, que incluye:

  a) Acceso, protección y administración de usuarios.
  b) Comunicación entre usuarios y entre procesos.
  c) Manipulación de E/S y administración de periféricos.
- Supervisión de la transmisión de datos entre la memoria principal y los dispositivos periféricos

El núcleo (kernel) reside en la memoria central y tiene el control de todo el sistema, por lo que ningun otro proceso lo puede interrumpir; sólo lo pueden llamar para que proporcione algún servicio de los ya mencionados conocidos como llamadas al sistema.

El kernel se divide:

Despachador: Este asigna recursos, programa procesos y atiende los requerimientos de servicio.
Manejador de interrupciones: Supervisa la transferencia de datos entre la memoria principal y los dispositivos periféricos.
Rutinas de apertura y cierre (Sincronización): Se logra por un mecanismo llamado evento es decir, los procesos esperan a que ocurran los eventos.

Cuando se inicia la operación en la computadora se carga en la memoria una copia del núcleo, que reside en el disco magnético (esta operación recibe el nombre de bootstrap). Para esto UNIX inicializa las interfases básicas de hardware que incluyen el reloj que proporciona interrupciones periódicas. El núcleo prepara algunas estructuras de datos, que incluyen una sección de almacenamiento temporal para transferencia de información entre terminales y procesos, una sección para almacenamiento de descriptores de archivos y una variable que indica la cantidad de memoria principal.

Despachador:

Debido a que existen pocas posibilidades de que haya espacio en la memoria para todos los programas simultaneamente, se asigna cierto número de ranuras en el disco, cada una capaz de contener una imagen del nucleo, es decir, una copia de tipo instantánea del contenido de memoria y de registros. Un proceso es la ejecución de una imagen, la cual contiene:

- Una imagen de almacenamiento.
- Valores generales de registro.
- El estado de los archivos abiertos.
- El directorio actual.

Una imagen del proceso reside en el almacenamiento principal durante la ejecución de ese proceso. Una imagen de almacenamiento está dividida en tres segmentos lógicos:

- Segmento de procedimiento reentrante (comienza en la localidad cero del espacio de direcciones virtuales)
- Segmento de datos.
- Segmento de pila.

Los segmentos de procedimiento pueden ser compartidos entre los procesos; en la memoria primaria se mantiene una copia de un segmento compartido. El segmento de datos comienza después de un segmento de procedimiento y puede crecer hacia direcciones de almacenamiento más altas (hacia arriba). El segmento de pila comienza en la dirección más alta del espacio virtual y crece hacia abajo, al recibir información por llamadas a subrutinas e interrupciones.

Los segmentos de textos (sólo de lectura del sistema), están controlados en la tabla de textos. Cada entrada tiene las direcciones de almacenamiento secundario del segmento y de almacenamiento primario. Cuando un proceso ejecuta por primera vez un segmento de texto compartido, ese segmento se coloca en el almacenamiento secundario, y se crea entrada en la tabla de textos con las direcciones de almacenamiento apropiadas y un contador indicando el número de procesos que comparten el segmento. Cuando este contador es cero, la entrada es liberada y los almacenamientos primario y secundario empleados por el segmento son desocupados.

El segmento de datos de la imagen de almacenamiento contiene datos de lectura-escritura privados de este proceso. Los datos del sistema asociados a este proceso se mantienen en un segmento separado de tamaño fijo. Este segmento de datos del sistema se intercambia con el proceso. El segmento del sistema contiene datos acerca de los procesos activos como:

- Registro de preservación.
- Descriptores de los archivos abiertos.
- Datos de contabilidad.
- Areas de datos de borrado.
- Pila para la fase de sistema del proceso.

Un segmento de datos del sistema no es directamente direccionable por el proceso al cual corresponde. Cada proceso tiene una tabla del proceso que contiene los datos requeridos por el sistema cuando el proceso no está activo. La tabla del proceso contiene el nombre del proceso, la localización de sus segmentos, e información de planificación.

Creación de procesos: Los nuevos procesos son creados por una primitiva del sistema llamada bifurnicación, la cual hace que el proceso actual se divida en dos procesos concurrentes llamados proceso padre y proceso hijo. Estos procesos no comparten el almacenamiento primario pero comparten todos los archivos abiertos. Se hacen copias para el hijo de todos los segmentos de datos que pueden escribirse. La llamada de bifurnicación devuelve el valor para que cada proceso pueda determinar si es el padre o el hijo.

Rutinas de Apertura y Cierre:

Para la sincronización de procesos, las tablas de procesos están asociadas a los eventos. Los eventos son representados como las direcciones de las tablas correspondientes. Un proceso padre en espera de que termine uno de sus procesos hijos, espera por un evento que es la dirección de su propia entrada de la tabla del proceso. Un proceso que termina, señala el evento representado por la entrada del padre en la tabla del proceso. Señalar un evento para el cual no hay ningún proceso en espera no tiene efecto.

Los procesos pueden ejecutarse en uno de dos estados: usuario o sistema. En el primero, un proceso ejecuta los programas del usuario y accede al segmento de datos del sistema. El objetivo principal de la planificación de procesos en un sistema UNIX es controlar los servicios para los usuarios interactivos. El despachador planifica segun prioridades. Las prioridades iniciales son asignadas por el código del nucleo que procesa los eventos-esperas. Los eventos de disco reciben una alta prioridad. Los eventos de terminal, eventos de hora y los de proceso del usuario reciben progresivamente prioridades bajas.

Manejo de interrupciones:

A los procesos del usuario se les asignan prioridades basadas en la cantidad de tiempo de procesador que han recibido. Todos los procesos del sistema tienen prioridad mayor que los procesos del usuario, por lo que siempre son servidos antes.

Los procesos pueden terminar voluntariamente por medio de la primitiva exit, o involuntariamente, como resultado de acciones, señales ilegales o trampas generadas por el usuario. Las trampas son usadas como referencias a direcciones incorrectas o intentos de ejecución de códigos de operación no definidos. La terminación involuntaria hace que la imagen del proceso se escriba en un archivo.

Esta imagen puede ser examinada por un programa de depuración que determine la razón de la terminación. La primitiva interrupt puede utilizarse para terminar un programa. La primitiva quit funciona como interrupt, pero tambien escribe el archivo imagen del proceso.

Entre las diferentes llamadas al sistema para manejo de procesos que existen en UNIX están las siguientes:

- **fork** (duplica un proceso).
- **exec** (cambia la identidad de un proceso).
- **kill** (envia una señal a un proceso).
- **signal** (especifica la acción por ejecutar cuando se recibe una señal de otro proceso).
- **exit** (termina un proceso).

En UNIX, las interrupciones son causadas por lo que se conoce como eventos, entre los cuales se consideran:

- La ejecución de una tarea de E/S.
- La terminación de los procesos dependientes de otro.
- La terminacion de la fraccion de tiempo de un proceso.
- La recepcion de una señal desde otro proceso.

En un sistema de tiempo compartido se divide el tiempo en un número de intervalos o fracciones y se asigna cada una de ellas a un proceso. UNIX toma en consideración además que hay procesos en espera de una operación de E/S y que ya no pueden aprovechar su fraccion. Para asegurar una buena distribución del procesador entre los procesos, se calculan dinámicamente las prioridades de los procesos para determinar cual será el proceso que se ejecutara cuando se suspenda el proceso activo actual.

Unix permite que los programas sean independientes de los dispositivos periféricos: la salida de cada programa o utileria del sistema pueden ser dirigidas a archivos en disco, impresoras o terminales y existe también la posibilidad de comunicación entre procesos para crear conjuntos arbitrarios y complejos de procesos concurrentes cooperativos.

La comunicación con la unidad central de procesamiento en el sistema UNIX es por medio del programa especializado de control llamado Shell. El shell es un lenguaje de control, un intérprete y un lenguaje de programación, tiene las siguientes características que lo hacen flexible para las tareas de un centro de computo:

a) Ofrece las estructuras de control normales: secuenciación, iteración condicional, selección.
b) Paso de parámetros.
c) Sustitución textual de variables y cadenas.
d) Comunicación bidireccional entre órdenes de Shell.

El shell permite modificar en forma dinámica las características con que se ejecutan los programas en UNIX.

Existen varios tipos de Shell con diferentes características:

- **BOURNE SHELL:** Es el intérprete de comandos básicos.
- **C-SHELL:** Es el intérprete de comandos creado en Berkeley para el sistema operativo BSD y para el XENIX, un poco más completo que el anterior. Su programación es en lenguaje C.
- **KORN SHELL:** Se basa en los dos anteriores, siendo compatible con Bourne 95%. Añade posibilidades de programación avanzada, facilidades aritméticas y mayor rapidez de ejecución.

3.4.2 Manejo de Memoria.

Dependiendo de la computadora en que se ejecute, UNIX utiliza dos técnicas de manejo de memoria: swapping y memoria virtual. Lo estandar en sistemas UNIX es un sistema de intercambio de segmentos de un proceso entre memoria primaria y memoria secundaria, llamado swapping, lo que significa que se debe mover la imagen de un proceso al disco si éste excede la capacidad de la memoria principal y copiar el proceso completo a memoria secundaria conforme sea necesario. Si un proceso necesita crecer, pide mas memoria al sistema operativo y se le da una nueva sección, lo suficientemente grande para acomodarlo. Entonces se copia el contenido de la sección usada al área nueva, se libera la sección antigua y se actualizan las tablas de descriptores de procesos. Si no hay suficiente memoria en el momento de la expansión, el proceso se bloquea temporalmente y se le asigna espacio en la memoria secundaria. Se copia a disco y, posteriormente, se devuelve a memoria principal cuando se tenga el espacio adecuado, lo cual sucede normalmente al cabo de algunos segundos.

El proceso que se encarga de los intercambios entre memoria y disco es llamado swapper, este jamas podrá perder su posición privilegiada en la memoria central. El núcleo se encarga de que nadie inteme siquiera interrumpir a este proceso, del cual dependen todos los demas procesos.

Cuando se decide traer a memoria principal un proceso en estado de "listo para ejecutar", se le asigna memoria y se copian allí sus segmentos. Entonces el proceso cargado compite por el procesador con todos los procesos cargados. Si no hay suficiente memoria, el proceso de intercambio examina la tabla de procesos para determinar cual puede ser interrumpido y llevado al disco.

Por otro lado cuando UNIX opera en máquinas más grandes, entonces utiliza el manejo de memoria de paginación por demanda. El tamaño de la página en UNIX es de 512 bytes en algunos sistemas, y de 1024 en otros. Para reemplazo se emplea un algoritmo que mantiene en memoria las páginas usadas más recientemente.

Como se menciona en el capítulo II, el sistema de paginación por demanda ofrece muchas ventajas en cuanto a flexibilidad y agilidad en la atención concurrente de múltiples procesos; además proporciona la capacidad de trabajar con procesos de tamaño mayor que el de la memoria central.

3.4.3 Manejo de Entradas y Salidas:

El sistema de entrada/salida se divide en dos sistemas complementarios:

- Sistema de E/S estructurado por bloques: Este se emplea para el manejo de discos y cintas magnéticas, emplea bloques de tamaño fijo de 512 o 1024 bytes para leer o escribir. Su propiedad esencial consiste en que es posible leer o escribir cada bloque en forma independiente de los demas; en otras palabras el programa, en cualquier momento, puede leer o escribir en cualquiera de los bloques. Los discos son dispositivos de bloques.
- Sistema de E/S por caracteres: Este se emplea para la atención a terminales, líneas de comunicación e impresoras; funciona byte por byte. Un sistema de E/S por caracteres entrega o acepta un flujo de caracteres, sin importar la estructura de bloque de que se trate; este no es direccionable y no tiene ninguna operación de localización.

UNIX emplea programas especiales conocidos como drivers para atender a cada familia de dispositivos de E/S. Los procesos se comunican con los dispositivos mediante llamadas a su manejador.

Para los procesos, los manejadores aparecen como si fueran archivos en los que se lee o escribe, logrando con esto homogeneidad y elegancia en el diseño.

Cada dispositivo se estructura internamente mediante descriptores llamados número mayor, número menor y clase (de bloque o caracteres). Para cada clase hay un conjunto de entradas, en una tabla, que apuntan a los manejadores de los dispositivos. El número mayor se usa para asignar el manejador correspondiente a una familia de dispositivos. El número menor del dispositivo pasa al manejador como un argumento, y este lo emplea para tener acceso a uno de varios dispositivos físicos semejantes.

Las rutinas que el sistema emplea para ejecutar operaciones de E/S están diseñadas para eliminar las diferencias entre los dispositivos y los tipos de acceso. No existe distinción entre acceso aleatorio y secuencial, ni hay un tamaño de registro lógico impuesto por el sistema. El tamaño de un archivo ordinario está determinado por el número de bytes escritos en él; no es necesario predeterminar el tamaño de un archivo. El sistema mantiene una lista de áreas de almacenamiento temporal (buffers), asignados a los dispositivos de bloques. El kernel usa estos buffers con el objeto de reducir el tráfico de E/S. Cuando un programa solicita una transferencia, se busca primero en los buffers internos para ver si el bloque que se requiere ya se encuentra en la memoria principal (como resultado de una operación de lectura anterior). Si es así, entonces no será necesario realizar la operación física de entrada o salida. La entrada/salida en un sistema UNIX se maneja principalmente en cinco llamadas al sistema: open, close, read, write y seek (para obtener y colocar información entre archivos y terminales, se usan otras tres llamadas gtty, stty y stat).

Para abrir un archivo se utiliza **fd = open (nombre del archivo, modo)** donde el modo indica si van a ejecutarse lectura, escritura o ambos, y fd es el descriptor de archivos que se utilizara en referencias posteriores al archivo.

La lectura y escritura se logran usando:

nobytesread = read (fd, buffer, nobytesdesired)

y

nobyteswritten = write (fd, buffer, nobytesdesired)

En el caso de una lectura, existen tres posibilidades, cada una de las cuales implica una lectura secuencial:

- Si esta es la primera lectura, entonces la lectura es secuencialdesde el principio del archivo.
- Si el archivo ha sido leido, entonces la lectura actual obtiene los datos que siguen a la anterior lectura de datos.
- Si acaba de realizarse una búsqueda, entonces la lectura es secuencial en el desplazamiento especificado en la llamada de busqueda.

Toda la lectura y escritura son secuenciales, pero el efecto de acceso directo se logra usando la llamada de búsqueda para ajustar el desplazamiento dentro del archivo como sigue:

seek (fd, desplazamiento, tipo_desplazamiento)

donde el tipo de desplazamiento especifica:

- si el desplazamiento es relativo o absoluto.
- si el desplazamiento es en unidades de bytes o en bloques de 512 bytes.

La operación de búsqueda funciona bien tanto en cinta magnética como en disco. Los archivos se cierran con sólo escribir:

close (fd)

Entrada/salida de flujo contra entrada/salida de registro.

Toda la entrada/salida de los sistemas UNIX es orientada hacia el flujo en vez de orientada hacia los registros, como en la mayoría de los sistemas. Un flujo es una secuencia de bytes, terminados por un delimitador (es decir, un carácter de fin de flujo). El concepto de flujo facilita la implementación de la independencia de dispositivos y la transparencia entre archivos, dispositivos y conductos. Otorga flexibilidad a los usuarios para tratar con colecciones de datos, pero supone una carga mayor para el usuario.

El usuario interesado en la entrada/salida de registro puede implementarla con bastante facilidad. Para implementar el procesamiento de registros de longitud fija, el usuario no necesita mas que especificar la longitud constante en todas las lecturas y escrituras. La entrada/salida de acceso directo con registros de longitud fija se logra multiplicando la longitud del registro por el numero de registro y llamando a busqueda, para situar el archivo en el registro deseado. Los registros de longitud variable pueden implementarse precediendo cada registro con un campo de tamaño de dos bytes, los registros de longitud variable pueden leerse por pares de lecturas como sigue:

read (fd, tamano, 2);
read (fd, buffer, tamano);

3.4.4 Manejo de Archivos e Información:

La estructura básica del sistema de archivos en UNIX es jerárquica, lo que significa que los archivos no están almacenados en un nivel sino en varios. Se puede tener acceso a cualquier archivo mediante su trayectoria, que especifica su posición absoluta en la jerarquía, y los usuarios pueden cambiar su directorio actual a cualquier posición. Existe un mecanismo de protección para evitar accesos no autorizados.

La distinción entre un directorio y un archivo ordinario es que el sistema se reserva el derecho de alterar el contenido de los primeros, y que el usuario solo los puede manipular mediante las órdenes mkdir y rmdir.

Los directorios contienen información para cada archivo, que consiste en su nombre y en un número que el kernel utiliza para manejar la estructura interna del sistema de archivos, conocido como el nodo-i. Hay un nodo-i para cada archivo, que contiene información de su dirección en el disco, su longitud, los modos de acceso, las fechas de acceso, el autor, etc. Existe, además una tabla de descriptores de archivos, que es una estructura de datos residente en el disco magnético, a la que se tiene
acceso mediante el sistema de E/S por bloques.

En Unix pueden existir varios sistemas de archivos independientes, y una misma unidad de disco magnético puede contener varios de ellos. Cada uno de los sistemas de archivos está dividido *internamente* en cuatro secciones o particiones lógicas:

1a. Bloque 0: Se reserva para procedimientos de bootstrap.
2da. Identificador 1:Contiene lo que se conoce como el "superbloque", que almacena un descriptor de la estructura de todo el sistema de archivos.
3a. Identificador 2: Es una lista de definiciones de archivos llamada lista-i (Esta es la tabla de archivos), en la que cada definición de archivo es una estructura de 64 bytes nodo-i. El desplazamiento de un nodo-i particular dentro de la lista-i es el número-i de un archivo, y actúa como su índice. La combinación del nombre del dispositivo y su número en esta lista sirven para identificar en forma única todo archivo.
4a. El final de cada sistema de archivos se usa para almacenar el contenido de los archivos, y es la sección de mayor tamaño.

El control del espacio libre en el disco se mantiene mediante una *lista ligada* de bloques disponibles. Cada bloque contiene la dirección en disco del siguiente bloque en cadena. El espacio restante contiene las direcciones de grupos de bloques del disco que se encuentren libres. De esta forma, con una operación de E/S el sistema obtiene un conjunto de bloques libres y un apuntador para obtener más.

Un nodo-i contiene 13 *espacios* para direcciones (de 4 bytes de longitud cada uno), en los cuales se encuentra la localización de un archivo. Las primeras 10 direcciones apuntan directamente a los primeros 10 bloques del archivo. Esto es suficiente para describir un archivo de hasta 10 x 512 bytes de longitud. Si el archivo es más grande, entonces se emplea la dirección 11, que apunta a un bloque que contiene hasta 128 direcciones de bloques adicionales. Es decir, el acceso a un byte situado entre la posición 5121 y la posición 70 656 (o sea, 512 x (10 + 128)) requiere de un acceso indirecto adicional para averiguar su posición exacta. Si es necesario un archivo aún mayor, entonces la dirección 12 apunta a un bloque de doble indirección con 128 bloques indirectos, donde cada uno apunta a 128 bloques del archivo. Por último, la dirección 13 apunta a un bloque de triple indirección, lo que permite un tamaño máximo para un archivo de 1 082 201 087 bytes.

Las operaciones de E/S en archivos se llevan a cabo con la ayuda de la correspondiente entrada del nodo-i en la tabla de archivos del sistema. El usuario normalmente desconoce los nodos-i y los números-i porque las referencias se hacen por el nombre simbólico de la trayectoria. Los procesos emplean internamente funciones primitivas (llamadas al sistema) para tener acceso a los archivos; las instrucciones más comunes son: open, creat, read, write, seek, close y unlink.

Toda estructura física se maneja "desde afuera" mediante la filosofía jerárquica de archivos y directorios, en forma totalmente transparente para el usuario. Desde el punto de vista del usuario hay tres clases de archivos:

a) Ordinarios.
b) Directorios.
c) Especiales.

a) Archivos ordinarios: se usan para almacenar información. Puede contener un programa, el texto de un documento, los registros de una compañía, o cualquier tipo de información que desee procesar en una computadora. El sistema no presupone una estructura particular en un archivo, sino que deja a los programas de los usuarios la tarea de manejar y controlar su estructura.

b) Archivos Directorios: Estos proporcionan la liga entre los nombres de los archivos y los archivos mismos; es decir, determinan una estructura en el sistema de archivos. En UNIX es responsabilidad de los usuarios la formación de su estructura arborecente en particular, pero el sistema operativo es el unico que puede alterar internamente el contenido de un directorio.

c) Archivos Especiales: Reside el medio de control sobre los dispositivos de E/S. Cada dispositivo esta asociado con al menos uno de esos archivos, que se leen y se escriben como si fueran un archivo ordinario de disco, pero en realidad son solicitudes de lectura y escritura activen el dispositivo asociado. Los archivos especiales no contienen información, sino que son utilizados para proporcionar un canal conveniente para los mecanismos de E/S. Existe un directorio dedicado a contener los archivos especiales (/dev). Para grabar informacion sobre una cinta magnetica, por ejemplo, se escribe en el archivo /dev/mt. Existen archivos especiales para cada linea de comunicación, cada disco, cada unidad de cinta y para la memoria principal física.

Las ventajas de tratar a los dispositivos E/S de esta forma son múltiples: un archivo y un dispositivo de E/S se vuelven similares; los nombres de archivos y de dispositivos tienen la misma sintaxis y significado, así que a un programa que espera un nombre de archivo como parámetro puede dársele un nombre de dispositivo con lo que se logra interacion rápida y fácil entre los procesos de alto nivel. Por último, los archivos especiales estan sujetos al mismo mecanismo de protección de los archivos regulares.

Modo de Protección:

El modo de protección consiste en asignar a cada archivo el número único de identificación de su dueño, junto con 9 bits de protección que especifican permisos de lectura, escritura y ejecución para el propietario, y para otros usuarios. Antes de cualquier acceso se verifica su validez consultando estos bits, que residen en el nodo-i de todo archivo. Ademas de lo anterior existen otros tres bits que se emplean para manejos especiales relacionados con la clave del supervisor (superusuario).

Las principales instrucciones para acceso interno al sistema de archivos son entre otras:

- open .	Convierte un nombre simbólico del sistema de archivos
-trayectoria-	en una entrada a la tabla de nodos-i.
create.	Crea una nueva entrada en la tabla de nodos-i.
- read y write.	Transferencia de un número determinado de bytes entre un archivo y la memoria.
- seek.	Permite acceso aleatorio dentro de un archivo.
- close.	Libera las estructuras creadas por open y creat.
- unlink.	Elimina un archivo del sistema.

El acceso inicial a un archivo es mediante las instrucciones open o create. Ambas devuelven un número conocido como descriptor de archivo, que sirve como conector entre el archivo y las llamadas de E/S del programa (un numero negativo indica un error de algun tipo).

Otra característica de Unix es que no requiere que el conjunto de sistemas de archivos resida en un mismo dispositivo. Es posible definir uno o varios sistemas desmontables que residen físicamente en unidades de disco diversas. Existe una orden (mkfs) que permite crear un sistema de archivos adicional, y una llamada al sistema (mount) con la que se añade (y otra con la que se quita) uno de ellos al sistema de archivos global.

El control de impresoras de una computadora que opera con el sistema operativo UNIX es mediante un subsistema (SPOOL) que se encarga de coordinar los pedidos de impresión de múltiples usuarios. Existe un proceso del kernel (/usr/lib/lpd) que periodicamente revisa las colas de servicios de las impresoras para detectar la existencia de pedidos e iniciar entonces las tareas de impresión. Este tipo de procesos, que son activados en forma periódica por el núcleo del sistema operativo reciben en UNIX el nombre de daemons (duendes) tal vez porque se despiertan y aparecen sin previo aviso.

CAPITULO 4.

Propuesta de programa para la impartición de la materia de Sistemas Operativos en la carrera de MAC.

4.1 Crítica al actual temario.

La materia de Sistemas Operativos tiene un carácter de optativo para las preespecialidades de *Sistemas Computacionales* e *Ingeniería* en la carrera de MATEMATICAS APLICADAS Y COMPUTACION.

Tiene como antecedentes Ensambladores y Estructuras de Datos y ninguna materia como consecuente; está diseñada para ser impartida en 80 horas semestrales de las cuales 48 horas deberan ser tróricas y 32 prácticas.

El objetivo principal de la materia es:

" El alumno desarrollara un Sistema Operativo o un monitor para un sistema PDP-11".

Dado que el estudiante no tiene conocimientos previos de Sistemas Operativos, considero que 80 horas no son suficientes para alcanzar el objetivo principal de la materia. Además de que el sistema PDP-11 actualmente se encuentra fuera del mercado de computadoras lo que hace muy difícil la construcción del mismo. Aunado a las limitaciones de equipo e infraestructura de la escuela para realizar esta tarea.

Las unidades temáticas de actual temario son:

I . Aplicaciones del Ensamblador.
II . Organización y generación de un Sistema Operativo.
III. Manejo de Entrada y Salida.
IV . Memoria virtual y Administración de memoria.
V . Desarrollo de un sistema de monitor.

Debido a que los objetivos de las unidades temáticas del programa no están claramente definidos y los contenidos de las unidades anteriores no existen en el programa actual, provoca que cada profesor de la materia interprete a su manera los contenidos, lo cual tiene serias consecuencias en la preparación de los estudiantes, ya que muchas veces dicha interpretación no cubre un estudio satisfactorio de la materia.

Ninguna unidad temática propone aplicar los conocimientos adquiridos a un caso específico o Sistema Operativo en particular.

Finalmente la bibliografía sugerida por el temario no se encuentra en la biblioteca de la escuela y algunos de los títulos propuestos son tan viejos que ya no se publican actualmente y además son difíciles de conseguir en alguna biblioteca.

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ACATLAN

CARRERA: MATEMATICAS APLICADAS Y COMPUTACION
DIVISION DE: CIENCIAS BASICAS
DEPARTAMENTO DE: MATEMATICAS APLICADAS

PROGRAMA BASICO DE LA MATERIA: SISTEMAS OPERATIVOS

CARACTER: OPTATIVO PARA LA PREESPECIALIDAD EN INGENIERIA

UBICACION:	8º	SEMESTRE	**VALOR:**	8	CREDITOS
HORAS SEMANALES:	3	TEORICAS		2	PRACTICAS
HORAS SEMESTRALES:	48	TEORICAS		32	PRACTICAS

MATERIA ANTECEDENTE: ENSAMBLADORES, ESTRUCTURA DE DATOS
MATERIA CONSECUENTE: NINGUNA

OBJETIVO GENERAL DESARROLLARA UN SISTEMA OPERATIVO O UN MONITOR PARA UN SISTEMA PDP-11.

UNIDAD TEMATICA	OBJETIVO DE LA UNIDAD TEMATICA
I. APLICACIONES DEL ENSAMBLADOR. (15 horas)	Aplicaciones particulares del ensamblador.
II. ORGANIZACION Y GENERACION DE UN SISTEMA OPERATIVO (15 horas)	Aplicará los fundamentos particulares y procedimientos de la organización y generación de un sistema operativo.
III. MANEJO DE ENTRADA Y SALIDA. (18 horas)	Aplicará los conceptos de niveles de interrupción para entrada y salida.
IV. MEMORIA VIRTUAL Y ADMINISTRACION DE MEMORIA. (12 horas)	Analizará diferentes tipos de memoria virtual y de memoria auxiliar. Aplicación de técnicas de estructura de datos en memoria. Manejo de particiones físicas y lógicas.
V. DESARROLLO DE UN SISTEMA MONITOR. (20 horas)	Desarrollará un sistema monitor para el sistema PDP-11

BIBLIOGRAFIA DE LA MATERIA: **SISTEMAS OPERATIVOS**

BIBLIOGRAFIA BASICA

D.E.C. — Rsx-11 M. Beginner's Guide, TEWKSBURY, MASS., 1979.

D.E.C. — Introduction to Rsx-11 M, TEWKSBURY, MASS, 1979.

KATZAN, H. — Operating Systems: A progmatic Approach, VAN NOSTRAND-REMHOLD NEW YORK, 1973.

GRAHAM, R.M. — Principles of Systems Programming, ADDISON-WESLEY READING, MASS, 1975.

BIBLIOGRAFIA COMPLEMENTARIA

WILKES, M.V. — Time Sharing Computer Systems, AMERICAN ELSEVIER NEW YORK, 1972.

SHAW, A. — The Logical Design of Operating Systems, PRENTICE-HALL ENGLEWOOD CLIFFS, N.J., 1974.

Una vez analizado lo anterior y considerando que actualmente se realiza la actualización del plan de estudios de la carrera de MAC mi propuesta es la siguiente:

1. La impartición de la materia de Sistemas Operativos como materia obligatoria para todo alumno que estudie la carrera de Matemáticas Aplicadas y Computación, debido a la importancia del tema para la formación del mismo.

2. Proporcionar conceptos sólidos de lo que es un Sistema Operativo. Para esto el alumno deberá tener conocimientos previos de Datos y Estructuras de Almacenamiento para comprender fácilmente el concepto de un sistema operativo.

3. Dar un carácter teórico-práctico a la materia, ya que considero que es importante que el alumno adquiera los fundamentos y las herramientas para poder manipular cualquier Sistema Operativo.

4. Una vez comprendido lo que es un Sistema Operativo así como sus componentes y el funcionamiento de los mismos, el estudiante será capaz de manipular cualquier sistema operativo que se encuentre en el mercado. Para efectos prácticos del curso el profesor elegirá un sistema operativo.
La justificación a este punto se basa en la rapidez con que se vuelven muchas veces obsoletos los sistemas operativos en el mercado y a la importacia que tiene el que los temarios de la carrera puedan estar actualizandose sin perder su contenido esencial.

A continuación presento la propuesta al oficial de la materia:

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
"A C A T L A N"
COORDINACION GENERAL DE ESTUDIOS PROFESIONALES

CARRERA DE: MATEMATICAS APLICADAS Y COMPUTACION

PROGRAMA DE MATERIA

ASIGNATURA: SISTEMAS OPERATIVOS CLAVE:

DIVISION: MATEMATICAS E INGENIERIA PROGRAMA: ACTUARIA Y MAC

CARACTER: OBLIGATORIO UBICACION: TERCER SEMESTRE

H O R A S

CLASIFICACION	SEMANA	SEMESTRE
T E O R I C A S	3	50
P R A C T I C A S (Laboratorio, Taller o Prácticos Externos)	2	30
T O T A L	5	80
C R E D I T O S		8

S E R I A C I O N

MATERIA ANTECEDENTE DATOS Y ESTRUCTURAS DE ALMACENAMIENTO
MATERIA CONSECUENTE ANALISIS Y DISEÑO DE SISTEMAS
REQUISITOS

O B J E T I V O G E N E R A L

El alumno AL TERMINO DEL CURSO, TENDRA LOS CONOCIMIENTOS BASICOS DE LAS PRINCIPALES FUNCIONES Y COMPONENTES DE UN S.O., SERA CAPAZ DE COMPRENDER Y EXPLICAR LA FORMA EN QUE OPERA Y EL PORQUE DE DETERMINADAS SALIDAS DE UN PROCESO

A C T U A L I Z A C I O N

FECHA	PARTICIPANTES
JUNIO '93	LIC. SARA CAMACHO CANCINO
	MARITZA NOVA JUAREZ

T E M A T I C A.

TEMAS Y SUBTEMAS	OBJETIVO ESPECIFICO	HORAS
TEMA 1 INTRODUCCION. Subtemas, - Breve Historia - Conceptos, funciones y caracteristicas de un Sistema Operativo. - Tipo de Sistemas Operativos.	Definirá el concepto y funciones de un Sistema Operativo.	10
TEMA II Componentes de un Sistema Operativo Subtemas. - Núcleo - Despachador - Manejo de interrupciones - Rutinas de Apertura y Cierre - Administración de Memoria - Administración de memoria contigua simple. - Administración de memoria por particiones. - Administración de memoria por particiones relocalizables. - Memoria Virtual - Administración de memoria paginada. - Administración de memoria por demanda. - Administración de memoria segmentada. - Administración de memoria segmentada páginada. - Manejo de información. - Gestión de archivos. - Gestión de trabajos. - Manejador de Entradas/Salidas. - Interfaz del Usuario.	Explicará el funcionamiento de un Sistema Operativo analizando la forma en que apoyen los componentes del mismo.	50
TEMA III Caso de Estudio	Aprenderá a manejar un Sistema Operativo Actual	20

GENERAL

1) **Bach Maurice.** "The Design of The Unix Opering System", 1986- Prentice Hall, New Jersey.

2) **Barron David.** "Sistemas Operativos (Para micros, minis y macrocomputadoras)". 1985 Mc Graw-Hill.

3) **Ben-Arir** "Principales of Concurrent Programming", 1982 Prentice Hall International, Londres.

4) **Irwin, Richard** "Principles of data processing concepts, applications and cases". 1986 2da. edición Ed. Prentice-Hall.

5) **Duffy Tim** "Four software Tools plus 1984. E.d. Wads wort Inc. U.S.A.

6) **Harrey M. Deitel** "An introduction to operating systems". 1988, Addison-Wesley Publishing Company.

7) **Levine Guillermo** "Introducción a la computación", 1989 2da. edición -- Mc Graw-Hill.

8) **Madnick, Stuart & John Donovan** "Operating Systems". 1974 Mc Graw-Hill, New York.

9) **Salas Parrilla Jesus** "Sistemas Operativos y Compiladores". 1991 1a. -- edición, Ed. Mac Graw-Hill.

10) **Tanenbaum Andrew** "Sistemas Operativos (Diseño e Implementación). 1988 2da. edición Ed. Prentice Hall.

Conclusiones

La carrera de Matemáticas Aplicadas y Computaciòn surge como una respuesta a la necesidad de contar con profesionistas capaces de resolver problemas sociales por medio de la computadora y dado que la computaciòn es un ciencia que cambia constantemente. Los programas de estudio deben actualizarse en forma periodica.

La competencia en el ámbito laboral a la que se enfrenta el egresado de MAC exige tener conocimientos de todas las novedades computacionales, lo cual no es fácil debido a los cambios constantes, la amplitud de temas y la falta de recursos materiales (equipo de cómputo).

La naturaleza de las materias relacionadas con la computación me permitió sugerir una solución a la problemática actual. La cual consiste en proporcionar planes de estudios semiabiertos, es decir planes que contengan una parte teórica básica de la materia y otra parte que permita mantener al alumno actualizado. Esta parte se deja a criterio del profesor quien tendrá el compromiso de transmitir su experiencia profesional enfocada al desarrollo actual de la materia, tal es el caso de Sistemas Operativos.

El temario propuesto de esta materia permitira al alumno adquirir los conocimientos basicos de la materia y las herramientas para aprender facilmente cualquier sistema operativo que requiera manejar el futuro ya sea por necesidad escolar o laboral. Ademas de aprender un sistema operativo vigente en el mercado que sera seleccionado por el profesor segun su criterio.

En este trabajo en caso, de que el alumno tuviese que realizar un examen extraodinario debe encontrar todo lo relacionado a los temas generales de la materia, y deben ser suficientes para evaluar los conocimientos de los alumnos sobre la materia.

APENDICE

FUNCIONES PRINCIPALES DE UNIX

En el presente apéndice se muestra un compendio de las funciones más importantes que incluye el sistema operativo UNIX. En general, cada una de las órdenes cuenta con un conjunto de opciones extra de procesamiento.

Funciones para control de usuarios:

login	Manejo de la conexión de un nuevo usuario y preparación de su entorno de trabajo en la terminal.
newgrp	Cambios de grupos de usuario.
passwd	Manejo y cambio de las claves secretas de acceso.

Manejo de Terminales.

stty	Determinación de opciones específicas de la terminal.
tab	Manejo generalizado de posiciones de tabuladores.

Manejo de Archivos y Directorios.

cat	Concatenación de uno o varios archivos. Despliegue en pantalla.
cd	Cambio de directorio de trabajo.
chmod	" " "
chown	" " "
chgrp	Cambio de permisos y facilidades de acceso a archivos y directorios.
cmp	Comparación de archivos e informe de diferencias.
cp	Copia de archivos o grupos de ellos.
dd	Traductor automático de formatos entre archivos.
find	Búsquedas estructuras de archivos, de acuerdo con criterios de fecha de creación, patrones de letras en los nombres o combinaciones lógicas.
ln	Liga de archivos entre sí.
mkdir	Creación de nuevos directorios.
mv	Cambio de nombres de archivos o directorios.
pack	Compactación de archivos para ahorrar espacios en disco.
pcat	Función idéntica a cat, para archivos compactados.
pr	Impresión de archivos paginados y con fecha.
rm	Eliminación de archivos.
split	Fraccionamiento de un archivo en partes.
tail	Despliegue de las últimas n líneas de un archivo.
unpack	Regreso de un archivo compactado a su estado original.

Ejecución de Programas.

echo	Escritura de mensajes por pantalla.
kill	Terminación de la ejecución de un proceso.
line	Lectura de órdenes por pantalla.
nice	Cambio de las prioridades de ejecución de órdenes y procesos.

sh	Intérprete del lenguaje de control.
	Redireccionamiento de entradas/salidas.
	Interconexión de procesos(pipes).
	Inicio de procesos por lotes (batch).
	Manejo de listas de argumentos y de variables de control.
sleep	Suspensión de la ejecución de una orden durante un tiempo especificado en segundos.
tee	Paso de datos entre procesos en ejecución y copia de los resultados obtenidos.
test	Prueba y uso de condiciones del Shell.
wait	Espera de la terminación de un proceso asincrónico.

Funciones para el control de status.

date	Informe de la fecha, que el sistema calcula automáticamente.
df	Informe de la cantidad de espacio disponible en los sistemas de archivos.
du	Despliegue de un resumen de la utilización del disco.
file	Determinación del tipo de información que contiene un archivo.
ls	Lista en orden de los nombres de los archivos en el sistema del usuario, con diversos grados de detalle.
ps	Informe de la actividad de los procesos del sistema en ejecución, activos y suspendidos.
pwd	Identificación del directorio actual de trabajo.
tty	Identificación de la terminal en la que se está trabajando.
who	Informe de los usuarios conectados al sistema.

Mantenimiento y Respaldos.

cpio	Operación de los dispositivos de almacenamiento masivo.
dumrp	Respaldo automático, selectivo y total, del sistema de archivos de la computadora.
fsck	Despliegue y reparación del sistema de archivos de la computadora, sus ligas, bloques usados, consistencia, y tamaño.
mount	Asignación de un sistema de archivos a un disco.
restor	Recuperación del sistema de archivos.
su	Asignación temporal de permisos privilegiados de acceso al sistema de archivos.
sync	Terminación de las operaciones de E/S pendientes.
tar	Manejo de la unidad de cinta magnetica.
umount	Cancelación de la asignación hecha con mount.

Funciones para impresión.

lp	Spooler para control de impresoras y pedidos de impresión.
lpr	Manejo de pedidos de impresión.
lpstat	Despliegue de información sobre el sistema de colas de impresion.
pr	Paginación de un archivo.

Manejo de Información.

awk	Lenguaje para procesamiento de patrones en textos.
calendar	Servicio automatico de recordatorios de fecha.
comm	Identificador de lineas comunes en dos archivos ordenados.
diff	Comparador de archivos e informe de diferencias.
grep	Despliegue de los renglones de un archivo que satisfacen criterios de reconocimiento de patrones.
join	Combinación de archivos con registros con llaves idénticas.
sort	Ordenamientos de archivos ASCII, con múltiples opciones.
tr	Transliterador de caráctéres, de acuerdo con convenciones definidas por el usuario.
uniq	Informe de lineas duplicadas dentro de un archivo.

Auditoría del Sistema Informe de Actividades

accdisk	Informe de actividades de uso del disco magnetico.
acctcon	Inicio de operación del sistema de auditoria interna.
acctprcq	Informe de actividades por proceso.
sag	Informe grafico de la actividad del sistema en un período cualquiera.

Facilidades de comunicaciones

cu	Llamado a otro sistema UNIX. Interfaz automática con otra computadora remota.
mail	Envio de mensajes a uno o varios usuarios.(Sistema de correo electrónico).
mesg	Control sobre los mensajes recibidos en una terminal.
news	Despliegue de la información del dia.
uucp	Transferencia de archivos entre sistemas UNIX.
uulog	Espera automática hasta lograr la conexión remota.
uuname	Informe de estadisticas de uso remoto.
uupick	Definición y control de subredes UNIX.
uustat	Transferencias entre dos máquinas remotas.
wall	Mensajes a todos los usuarios, por parte del administrador del sistema.
write	Comunicación directa entre terminales.

Herramientas de desarrollo de Programación.

adb	Depurador interactivo.
ar	Creación y mantenimiento de bibliotecas de programas en cinta magnetica.
as	Ensamblador.
ld	Editor-ligador de uso general.
library	Bibliotecas comunes de tiempo de ejecucion.
lorder	Manejo y ordenamiento de archivos objetos para ser cargados.
make	Sistema general para control y mantenimiento de programas, rutinas y módulos. Manejo automático de las interdependencias entre módulos.
od	Despliegue de códigos objeto en forma octal, hexadecimal, decimal y ASCII.
prof	Construccion de una tabla de estadisticas llamadas a funciones, rutinas y tiempos de ejecución (profiler).
size	Informe de los requerimientos de memoria de archivos objeto.
strip	Minimizacion del espacio requerido por un archivo objeto.
time	Ejecución de una orden del sistema y reporte de los tiempos de proceso y ejecución.

Lenguaje C.

cb	Embellecedor de programas en C.
cc	Compilador, ligador y cargador del lenguaje C. Macroprocesador integrado.
lint	Verificador sintáctico/semántico para el lenguaje C.

Otros lenguajes algorítmicos integrados.

bc	Interfaz tipo lenguaje C para la calculadora dc.
bs	Intérprete y compilador para un lenguaje, que comparte características de SNOBOL 4, BASIC y C.
dc	Calculadora programable interactiva de precisión aritmética ilimitada y manejo de múltiples bases numéricas.
snobol	Interprete y compilador del lenguaje SNOBOL.

Macroprocesamiento.

m4	Macroprocesador de propositos generales.

Preparación de Documentos.

troff	
nroff	Sistemas completos de procesamiento de palabras y tipografía computarizada.
deroff	Eliminación de las ordenes para el procesador troff en un texto.
ed	Editor interactivo de línea, guiado por gramática regular; incluye reconocimiento de patrones.
eqn	Preprocesador para diseño de expresiones matemáticas en troff.
nrm	Sistema de macros para nroff y troff.
marcheck	Verificación de los documentos a ser procesados por eqn y nrm.
ptx	Creación de tablas de índices permutados.
sed	Igual que ed, pero para archivos de tamaño ilimitado.
spell	Sistema de comparación automatica de palabras de un texto contra un diccionario, para encontrar errores mecanográficos.
tbl	Preprocesador para manejo y diseño de tablas en troff.
vi	Editor de pantalla, con facilidades integradas para creación de programas en lenguaje C.

Manejo de Gráficas.

graph	Producción de una gráfica a partir de sus coordenadas.
spline	Ajuste gráfico de curvas por metodos matemáticos.

BIBLIOGRAFIA

Bach Maurice. The Design of the Unix Operating System. 1986 Prentice Hall, New Jersey.

Barron David. Sistemas Operativos (Para micros, minis y macrocomputadoras). 1985. Ed. McGraw-Hill.

Ben-Ari, M. Principles of Concurrent Programming. 1982. Prentice Hall International, Londres.

Brian W. Kernighan. The Unix programing enviroment 1984. McGraw-Hill Massachusetts.

D. Irwin, Richard. Principles of data processing Concepts, Applications and cases. 1986. 2da edición Ed. Prentice-Hall.

Duffy Tim. Four Software Tools Plus. 1984. Ed. Wadswort Inc. U.S.A.

Harvey M. Deitel. An introduction to Operating Systems 1988. Addison-Wesley Publishing Company.

Levine Guillermo. Introducción a la computación 1989. 2da edición McGraw-Hill.

Madnick, Stuart & John Donovan. Operating System. 1974. McGraw-Hill, New York.

Salas Parrilla Jesus. Sistemas Operativos y Compiladores. 1991. 1a edición Ed. McGraw-Hill.

Tanenbaum Andrew. Sistemas Operativos (Diseño e Implementación). 1988 2da edicióon. Ed. Prentice Hall.