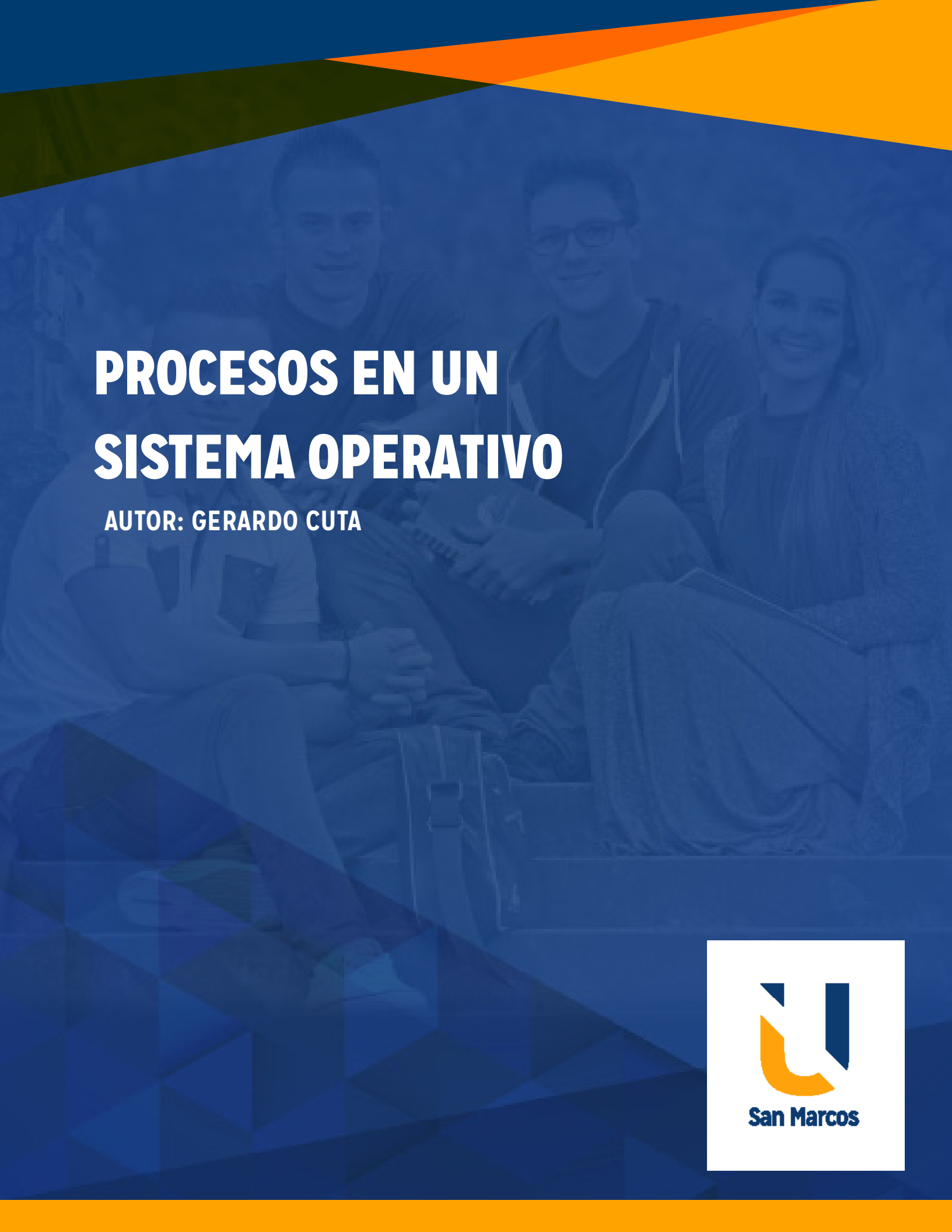
PROCESOS EN UN SISTEMA OPERATIVO

AUTOR: GERARDO CUTA



San Marcos

ÍNDICE

INTRODUCCIÓN

En esta sección se reconocerán los componentes que tiene un sistema operativo, ya que estableciendo sus particularidades se puede identificar el modo en que interviene en los componentes a su cargo; inicialmente desde el núcleo o su parte central, siendo este el ***software*** base que contiene los procedimientos básicos (Lenguaje de máquina) para la interconexión con los dispositivos de ***hardware*** dispuestos para el equipo en donde actúa el sistema operativo.

Adicional a esto, el sistema operativo permite realizar la gestión de diversos procesos entre los que se encuentran los de almacenamiento, ya sea primario o secundario, que a través de su ***software*** administra los diversos registros destinados para esta actividad. Al mismo tiempo, otro tipo de gestión se presenta en la ejecución del programa base del sistema operativo relacionado con el aspecto de los dispositivos de Entrada/Salida, administrando tanto los dispositivos que ingresan datos, como los que presentan la información de manera que el usuario pueda interpretarla.

Procesos en un sistema operativo

Para Muñoz (2012), los procesos pueden ser tareas, flujos de control o hilos, dependiendo del contexto que se está aplicando. Estos pueden ser concurrentes en un sistema, para lo que se debe contar con los recursos necesarios para soportarla, siendo asignada por la **CPU** del procesador. Teniendo en cuenta algunas prioridades que se presentan, sincroniza los procesos para que estos se ejecuten adecuadamente.

Cada vez que se ejecuta un programa, se establece una estructura de datos específica para cada proceso, lo que hace que se identifique cada programa y establece el correcto funcionamiento del mismo. A esta etapa se le llama Bloque de Control de Proceso o **BCP.**

Los procesos se pueden dividir en **HEBRAS**, entendidos como subprocesos o subrutinas que se utilizan para hacer eficiente la ejecución de un programa, ya que estas comparten recursos. Explicado de otra manera, cuando se activa un programa se habilitan ciertos recursos y ***hardware*** del equipo. En el caso de habilitar varios archivos del mismo programa, estos recursos seguirán habilitados para el mismo ***software***.

Concurrentes
Circunstancia de suceder o producirse varias cosas en un mismo momento.
Tomado de: http://es.thefreedictionary.com/concurrencia

¡Importante!

Los sistemas operativos actuales están desarrollados como multihebras, esto es, que varios programas se pueden encontrar habilitados simultáneamente y cada uno de ellos tiene sus recursos disponibles, aunque se encuentra ***software*** convencional que permite realizar las acciones secuencialmente.

En un proceso se pueden presentar varios estados, entre los que se encuentran en ejecución, estado en el que el programa está corriendo y se encuentra cargado en la CPU del procesador; también se puede presentar el estado de en espera, activo o preparando, a los cuales les falta un período de tiempo determinado para que sean ejecutados, encontrándose en un registro de la memoria caché del procesador; y por último, se puede encontrar **bloqueado**, cuando dos procesos utilizan el mismo fichero de datos.

Entonces, los procesos se encuentran identificados en los sistemas operativos, ya que cada uno de estos está siendo identificado por un indicador, también tiene un número que lo identifica para determinar su estado, el cual se conoce con el nombre de PID. Además de un proceso puede activar otro y se diferencian utilizando los términos de padre e hijo.

Kernel de un sistema operativo

El Kernel o núcleo de un sistema operativo, es un ***software*** base que tiene variedad de funciones, entre las que se encuentran facilitar a los diferentes programas el ingreso seguro al ***hardware*** de la PC por medio de los servicios de llamada de sistema, el cual establece el uso y la cantidad de tiempo que puede utilizar el ***hardware*** solicitado para realizar determinado proceso.

¡Importante!

Las funciones que el Kernel realiza específicamente están orientadas a diferentes tipos de gestión, entre las que se encuentran la de memoria, los procesos en ejecución y el tiempo que estos necesitan para ser implementados. También administra los programas que solicitan los recursos y las tareas detalladas de los equipos.

Tipos de Kernel

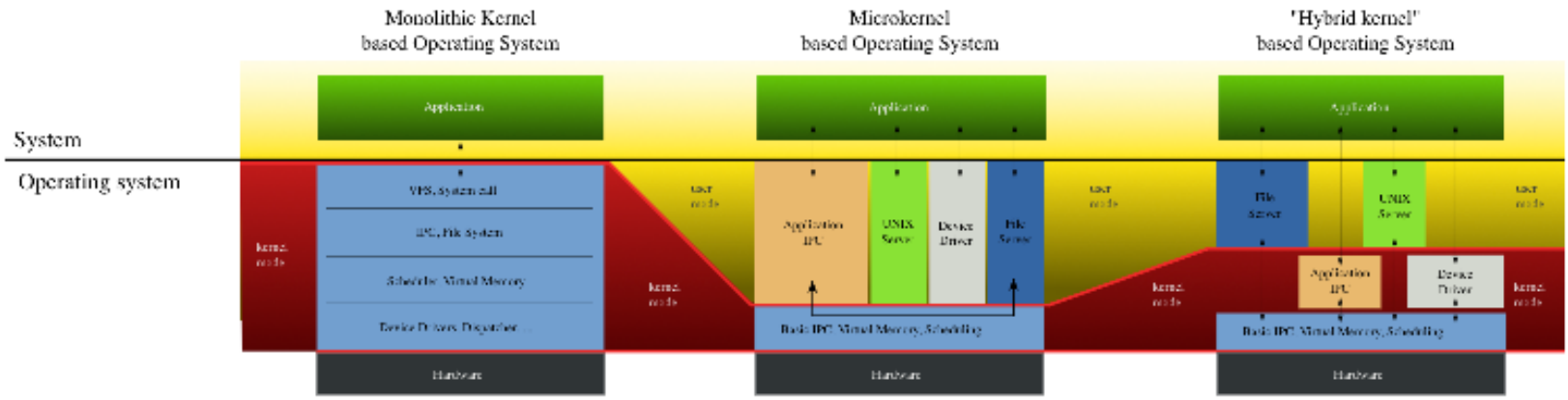


Figura 1. Tipos de Kernel
Fuente: By Golftheman - http://en.wikipedia.org/wiki/Image:OS-structure.svg, Public Domain, https://commons.wikimedia.org/w/index.php?curid=4397379

Inicialmente los programas se cargaban y se ejecutaban cada uno de manera independiente; para esto se debía instalar, reiniciando el equipo para que el nuevo ***software*** se cargara y estuviera dispuesto a ejecutar, pero a través del tiempo se han establecido espacios de memoria que permiten almacenar diversos procesos, así sean varios, ejecutados simultáneamente, apoyados en programas auxiliares del sistema operativo, llamados el **cargador** y el **depurador;** o también, se pueden cargar en un espacio de memoria específicamente.

Se encuentran cuatro tipos de Kernel en los diferentes sistemas operativos:

Kernel monolítico: la estructura del sistema operativo que cuenta con este tipo de núcleo se encuentra diseñada de manera completa en un solo programa, ya que todos los componentes del núcleo tienen acceso a todas las estructuras de datos y rutinas. A través del tiempo se le han realizado modificaciones en donde en estado de ejecución se puede realizar una carga dinámica de algunos módulos ejecutables, ubicándose específicamente en la memoria de núcleo llamado anillo 0**.** Se debe tener en cuenta que si se presenta un bloqueo de módulo también se bloqueará el núcleo. Este tipo de núcleo lo utilizan principalmente los sistemas operativos derivados de UNIX como LNUX y FreeBSD.

Anillo 0
En términos de sistemas operativos permiten el acceso a los diferentes recursos dispuestos.

Micro Kernel: este tipo de micronúcleo es utilizado en sistemas operativos que entrega un conjunto de llamadas mínimas para implementar los servicios básicos como espacio de direcciones, comunicación entre procesos y planificación básica. Este tipo de Kernel busca superar por si mismo errores de ***software*** o ***hardware***.

Planificación básica
Se produce cuando la CPU se encarga de asignar tiempos de ejecución a cada proceso según el tipo de algoritmo y prioridad en el proceso.

Lo que se busca con este tipo de núcleo es descentralizar fallos, creando y depurando controladores de dispositivos, lo que permite el aumento de la tolerancia a fallos y que las diferentes plataformas de ***hardware*** puedan ser portables. Este tipo de Kernel se encuentra principalmente en los diferentes sistemas operativos como Simbian, AIX, y Minix, entre otros.

Tolerancia a fallos
Es la capacidad de un sistema de almacenamiento para acceder a información específica o seguir en funcionamiento aun en caso de producirse algún fallo.

Kernel híbrido: este tipo de núcleo es un micronúcleo que contiene un código "específico y encriptado" que permite ejecutarse a si mismo con una mayor velocidad que si estuviera en un espacio de usuario. Los sistemas operativos que lo utilizan son Windows, Mac OS y XNU, que tienen un núcleo que no ha sido modificado.

Este tipo de núcleo tiende a ser confundido con núcleos monolíticos, que pueden cargar módulos después del arranque, realizando paso de mensajes y migración de código "no esencial" hacia el espacio del usuario manteniéndolo también en su propio núcleo.

Exo-Kernel: desarrollado en el Instituto Tecnológico de Massachusetts, este tipo de núcleo buscaba crear una capa de ***software*** superficial para otros sistemas virtuales, que tenía como objetivo hacer invisible los recursos de ***hardware*** y las aplicaciones que interactúan con este, lo que permitiría reducir tiempo en los procesos del Kernel y escribir los programas de una manera fácil. La intención del exonúcleo es el desarrollar paquetes de ***software*** que puedan ingresar al Kernel en el momento que sea necesario.

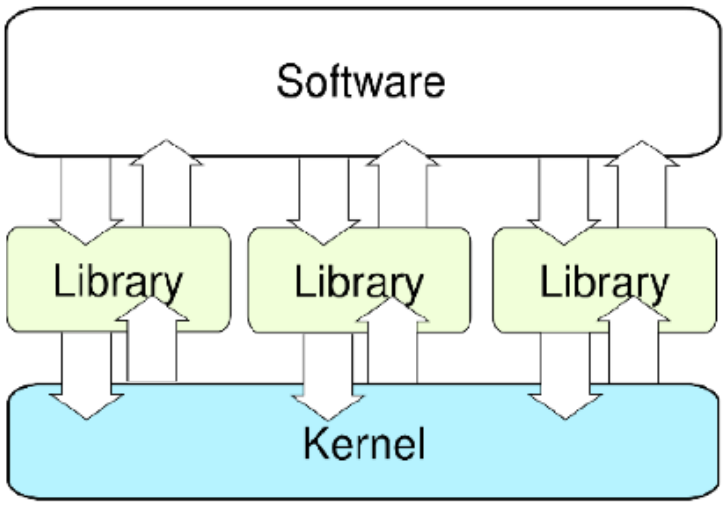


Figura 2. Esquema de ExoKernel
Fuente: By David Futcher - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4391572

Kernel de Linux

Este Kernel es el más complejo de todos, ya que su estructura tiene algunos defectos inherentes a los núcleos monolíticos que se deben tener en cuenta, pero los programadores de Linux buscan soportar estas limitaciones desde la implementación de módulos que pueden ser cargados y descargados dentro de la ejecución, recordando que a estos aditamentos se le deben reiniciar el proceso para que sean activados en el sistema operativo, aunque se están desarrollando ***software*** que se pueda instalar desde su ejecución.

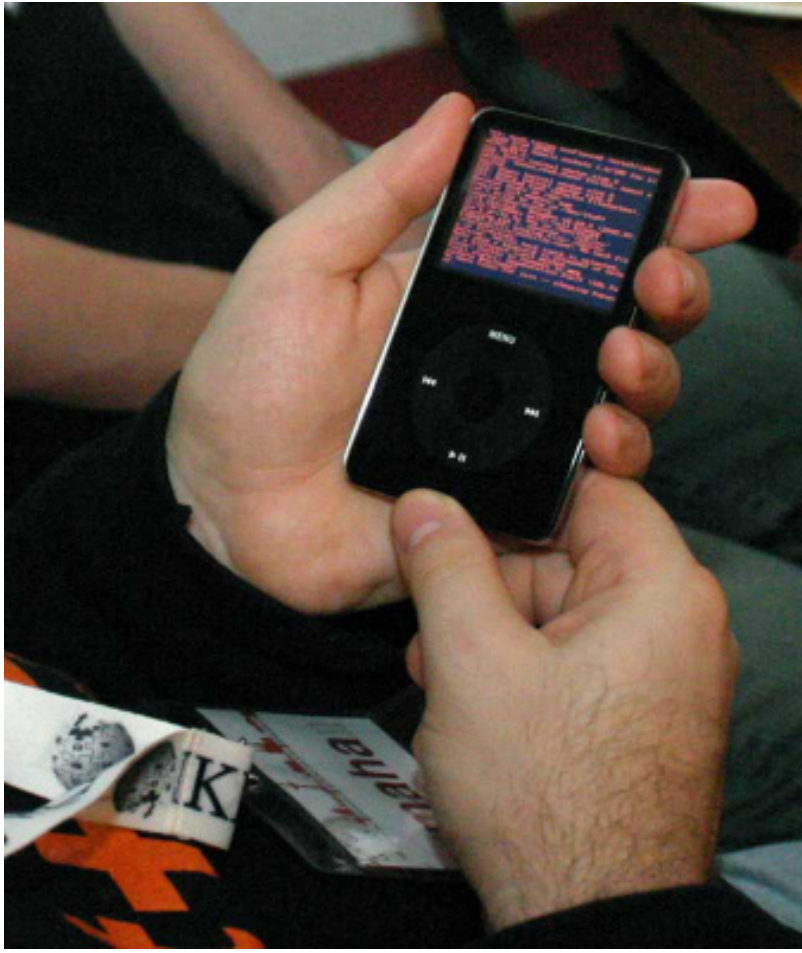


Figura 3. iPod ejecutando un núcleo Linux
Fuente: De Elke Wetzig (Elya) - Trabajo propio, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=1607578

Los módulos de Kernel para Linux llamados (LKM), permiten mantener en funcionamiento el núcleo con todo el ***hardware*** sin consumir la memoria disponible para este; se utilizan principalmente para cargar funciones como los sistemas de archivos y las llamadas al sistema, y se pueden actualizar durante el arranque con el comando menuconfig o cargando y descargando los módulos sobre la marcha con el comando modprobe, de acuerdo con EcuRed, (s.f).

LKM
Loadable Kernel Module (LKM) por sus siglas en inglés, es un módulo que permite extender el Kernel cando se encuentra en ejecución, con el se pueden agregar dispositivos, sistema de archivos o agregar llamadas al sistema.

Las empresas desarrolladoras de ***software*** no proporcionan su código fuente para insertar el código dentro del Kernel ya que son desarrollos propios, justificando esta decisión a la contaminación de código que se puede presentar al distribuir ***software*** libre.

Kernel de Windows

Este Kernel debe tener determinadas características en su estructura (recordando que este es modular) y se divide en ciertas partes, la primera es el **Modo Kernel**, que corresponde a una interfaz entre el resto del sistema, instalada en la carpeta de enlace dinámico (dll), protegiendo así al sistema operativo de otras particularidades que se encuentra en los programas para utilizar el ***hardware***, el cual se conoce con el nombre

de (HAL), y que debe ser modificado para cada tipo de procesador lanzado al mercado.

El **MicroKernel** es el administrador de todas las acciones que se presentan en el sistema operativo. Este planifica la ejecución de los hilos, y las interrupciones y excepciones del proceso, buscando mantener ocupado al procesador el máximo tiempo posible; sus funciones básicas están orientadas a la ejecución de subprocesos, la sincronización del multiprocesador y la gestión de interrupciones de ***hardware***. Este Kernel tiene un gestor específico para la administración de procesos, llamado de la misma manera, ya que genera, elimina y modifica estados de los procesos y proporciona la información necesaria en la memoria virtual del resto del sistema; este tipo de **memoria virtual** puede gestionar el uso del disco duro en vez de la memoria física para almacenar páginas de código, almacenándolas y entregándolas cuando el sistema lo necesita.

También participan otro tipo de elementos que establecen control dentro del sistema operativo, como **el administrador de Entrada/Salida (IN/OUT)** que permite la comunicación entre los distintos controladores del ***hardware*** instalado y el subsistema win32 que es el responsable de la (IN/OUT) de diversos dispositivos como mouse, teclado y monitor.

Gestión de memoria para los sistemas operativos

Según Muñoz (2012), para correr un programa en un ordenador, se debe partir del hecho de que normalmente, la cantidad de memoria que este necesita es mayor al espacio de almacenamiento primario que tiene físicamente el dispositivo que se encuentra dispuesto en la memoria RAM.

RAM (RANDOM ACCESS MEMORY)
Es una memoria que permite almacenar la información mientras se encuentra energizada y hace parte de la etapa de Almacenamiento en la Arquitectura del PC establecida por John Von Neumann.

Una de las soluciones dispuestas para tratar de resolver este inconveniente es el de dividir el programa en capas, lo que permite que la capa que sea necesaria para ejecutar el programa se cargue en la memoria principal y que el resto de capas se carguen en la memoria alterna o secundaria, en este caso el disco duro, ubicada en una zona especial llamada la memoria virtual, la cual tendrá los procesos del programa preparados para ser enviados a la memoria RAM en el momento que requieran las instrucciones. Esto asegura la existencia de espacio en la memoria principal para almacenar otros procesos que se estén llevando a cabo simultáneamente (como carga de video y ejecución de otros programas, entre otros.

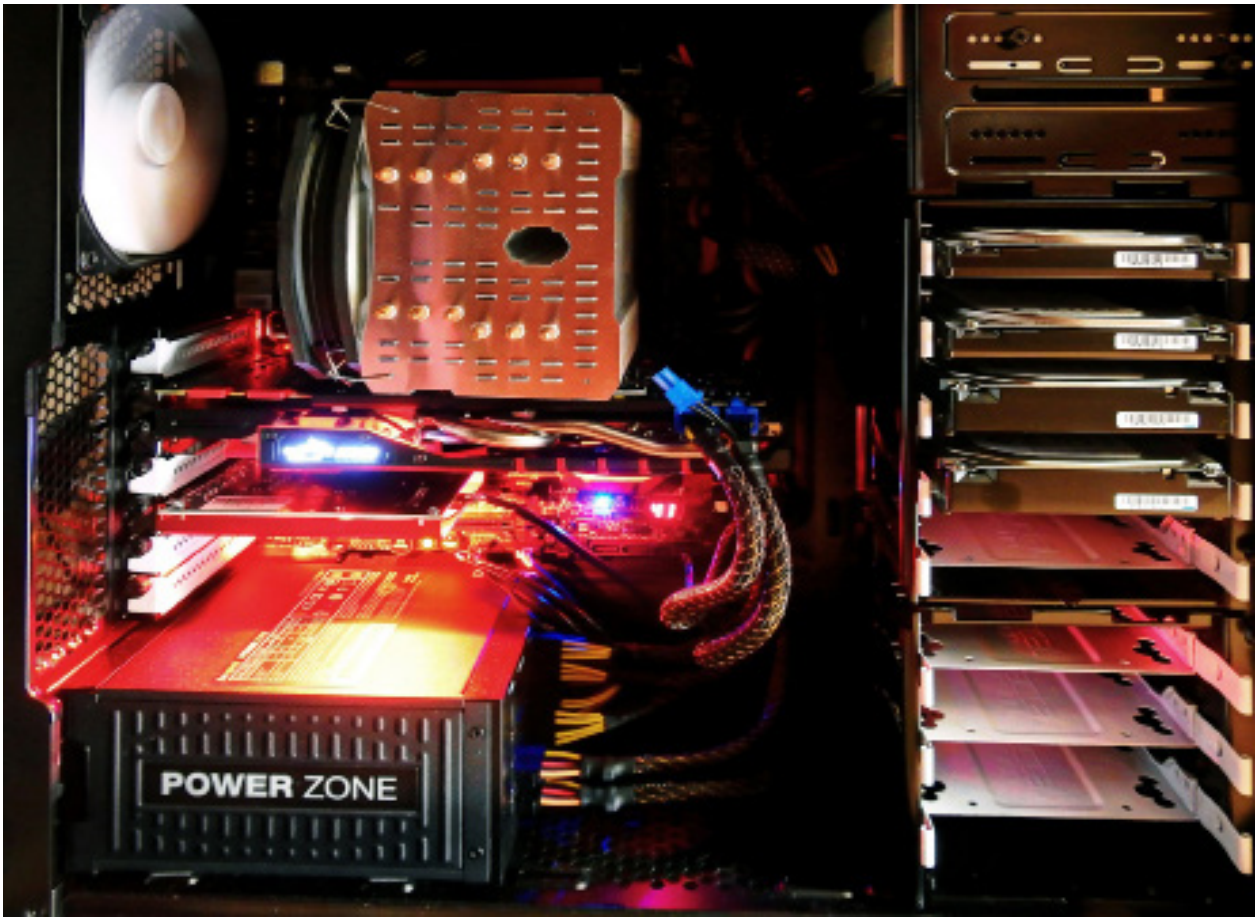


Figura 4. Gestión de memoria del PC
Fuente: https://pixabay.com/en/computer-technology-pc-electronics-1574533/

Este procedimiento se realiza principalmente en los sistemas operativos multitareas y multiusuarios, en donde se ha aumentado el uso de la memoria principal, siendo este uno de los factores relevantes que se tienen en cuenta en el momento de la instalación de un sistema operativo. Uno de los ejemplos más visibles es la gestión que realiza el sistema operativo Windows con todas sus versiones, pero se presenta un problema cuando se aplica este proceso, el espacio dedicado en la memoria virtual ubica la información de manera aleatoria, haciendo que la información se encuentre desordenada, haciendo más lenta la búsqueda de la información que se desea ejecutar. Esto no sucede en el sistema operativo Linux.

Entonces la administración de memoria tiene unas tareas asignadas, entre las que se encuentran el seguimiento de ubicación de la información en los espacios de memoria, ya sea principal (RAM), virtual o secundaria (HDD o SDD), planeando la ocupación de la información antes de que ocurra y liberando espacio que no necesita los programas o sus capas. También controla la transferencia de datos entre la RAM y el disco duro dependiendo del tamaño del proceso, ya que si ocupa un espacio considerable, se carga directamente en la memoria principal.

¡Importante!

La gestión multitarea en un sistema operativo es importante ya que cada una de las tareas asignadas debe encontrarse en un espacio específico llamado partición fija; por esto, los procesos deben solicitar ingresar a la memoria para preparar sus procesos e hilos que conllevan con anterioridad, para lo que se encuentra un planificador, almacenándose en el BCP. Una de las características que debe tener el planificador está dada por la cantidad de memoria que asigna a cada conjunto de instrucciones relacionadas a cada tarea, ya que al realizar un mal cálculo se puede perder espacio que más adelante no se podrá liberar ni desfragmentar, por este motivo se establecen algunas particiones variables.

BCP
Es el término utilizado para identificar el "Bloque de Control de Proceso).

Desfragmentación
Proceso por el cual se acomodan los archivos de un disco para que no se aprecien los fragmentos que se encuentran entre ellos.

Esto establece una labor adicional, determinar el espacio ocupado y libre para administrarlo en el tiempo necesario para dar respuesta a cada una de las tareas asignadas; para esto se utiliza la **paginación**, generando un procedimiento para reconocer el direccionamiento de estas a través de las **tablas de páginas**, las cuales se basan en dos aspectos específicos, tamaño de la tabla y el tiempo de asignación. Pero cuando la gestión de tabla no puede asignar memoria, pues los espacios pueden reducidos para esta tarea, se acude a la **segmentación**, referida a la asignación de espacios que no han sido ocupados por algún proceso, pero que su tamaño no favorece para ordenar una dirección de tabla específica, por lo que se acude a realizar una revisión del tamaño libre visto como segmentos, se evalúan sus espacios para ser asignados dependiendo del proceso a ejecutar.

Reflexionemos

Resumiendo, en la gestión de memoria se reconocen tres procesos principalmente; memoria virtual, gestión de tablas y segmentación, procesos complejos pero que permiten aprovechar al máximo el rendimiento de los dispositivos de almacenamiento en un equipo informático. Los sistemas operativos multitareas permiten la manipulación de estas características por parte del usuario, presentándose la desventaja de que si estas variables no son calculadas adecuadamente, afectan el rendimiento de la máquina.

Gestión de memoria en los sistemas operativos Windows y Linux

En estos sistemas operativos, la gestión de memoria cumple con los estándares anteriormente expuestos, presentando ciertas diferencias en la forma de organización del almacenamiento.

Linux

Este sistema operativo utiliza la memoria virtual como ya se había descrito anteriormente, pero la paginación se divide en tres niveles, primero se encuentra un directorio de páginas de modo general, en su segundo nivel está un directorio intermedio de páginas y en su último nivel está la tabla de paginación, permitiendo realizar una búsqueda de información localizada. Linux cuenta con un sistema de identificación de página que consiste en asignar un tamaño fijo y utilizando un algoritmo llamado de "Reloj", identificar el tiempo que ha transcurrido en ser utilizada cada página; si esta ya se ejecutó y se identifica que lleva un tiempo determinado, Linux considera en remplazarlo.

Windows

El gestor de memoria asigna un rango de tamaño de páginas que se encuentra entre los 4KB y los 64KB, estableciendo tres estados, el primero es el de disponible, en donde el gestor de memoria virtual identifica espacio en la memoria principal y permite acumular páginas en su espacio de almacenamiento, pero cuando considera que la memoria RAM se está llenando, se coloca en estado reservada, identificando las páginas que han sido utilizadas más tiempo y las asigna a ***swap*** para liberar espacio en memoria principal. Para eliminar páginas en Windows se realiza el mismo proceso de Linux, pero con la diferencia que solo lo hace cuando es necesario.

Gestión de entrada - salida de los periféricos en los sistemas operativos

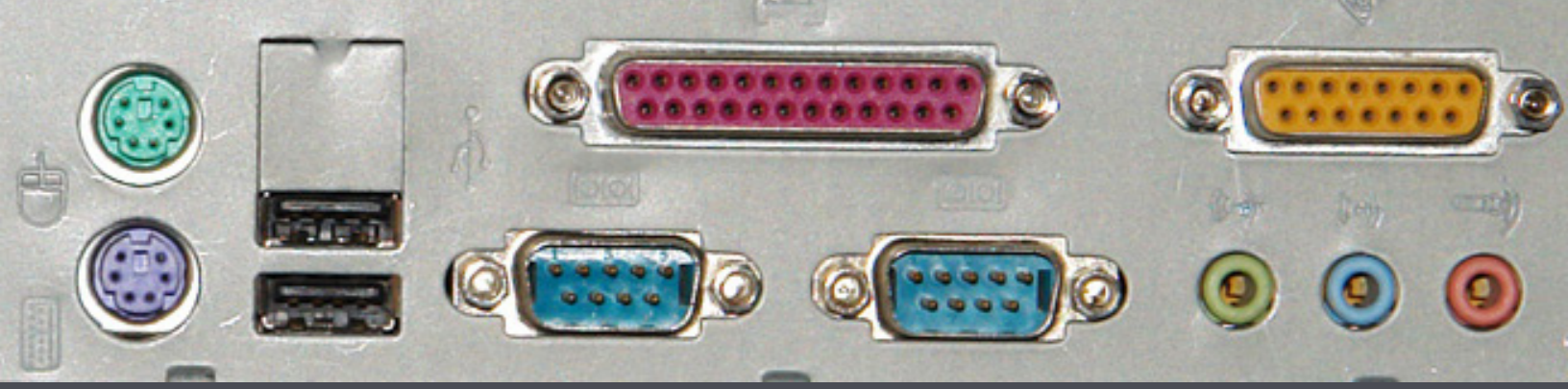


Figura 5. Puertos entrada/salida ordenador
Fuente: By George Shuklin - Own work, CC BY-SA 3.0, https://es.m.wikipedia.org/wiki/Archivo:Hardware_ports.jpg

Otra labor que debe desarrollar el sistema operativo es el control de los dispositivos de entrada y salida de la Información ***(In/Out)***, estableciendo prioridades para la interacción con el procesador y también identificando y corrigiendo errores que se puede presentar en esta etapa de la arquitectura del ***hardware***.

La clasificación dada a esta gestión no viene determinada por si los periféricos son de entrada o salida, sino que se establece por la forma de los datos trabajados, ya sean individuales cuando la información ingresa en forma de **caracteres**, sin ningún orden específico. El ejemplo más claro es el de los teclados, permitiendo a su vez evidenciar el caracter o cadena de caracteres desde los resultados obtenidos después del proceso, en dispositivos como el monitor o la impresora. También se encuentran los periféricos de **bloque**, que presentan la información de tamaño fijo, empaquetado, como la información que sale en bloque de los registros como resultado de ejecutar una orden dada por un proceso. Este se puede ver en los paquetes entregados a la memoria principal para que sean almacenados.

¡Recordemos!

Se debe recordar que los dispositivos periféricos son construidos como elementos de interfaz entre la máquina y el usuario; por este motivo, la tecnología de construcción de estos dispositivos están pensados para ser transductores de variables físicas (Sentidos del cuerpo humano), en pulsos eléctricos representados en binario para poder ser procesado y posteriormente almacenarse en la memoria principal o secundaria, dependiendo de la necesidad del cliente.

Transductores
Dispositivo que recibe un tipo de energía producida por la naturaleza y la transforma en pulsos de corriente directa (DC).

También se debe tener en cuenta que la interface de comunicación entre ***hardware*** y ***software*** se puede realizar de dos maneras, en la primera se cuenta con una interfaz tipo texto, en el que la comunicación se realiza en forma de cadena de caracteres. Tanto la manera como el usuario ingresa la información y la forma en que la recibe, así como ejemplos de sistemas operativos con este tipo de interface, se encuentra el Sistema Operativo de Disco (D.O.S) y UNIX. El otro tipo de interface que se presenta es el de gráficos, en donde la información se muestra en bloques que se representan por medio de iconos que, en el momento de acceder, ejecutan cierta cantidad de procesos sin necesidad de entregar órdenes por caracteres.

Se debe recordar que en el módulo de "Arquitectura de ***hardware***", se reconocen las diferentes clases de periféricos desde su manera de recibir y entregar la información, que puede ser de entrada o de salida. Existen dispositivos que permiten el ingreso y el egreso de la información como discos duros extraíbles, dispositivos móviles táctiles y tarjetas de red, entre otros. Otro aspecto a tener en cuenta es que estos dispositivos permiten potenciar las características de funcionabilidad del equipo informático.

El sistema operativo puede realizar la gestión de ***In/Out*** de varias maneras, la primera se conoce por muestreo, en donde el sistema operativo está realizando un seguimiento a los periféricos, utilizando los registros de entrada/salida, identificando qué dispositivos se encuentran conectados. Este control por muestreo se puede realizar de dos maneras, la primera se hace con **prioridad uniforme** en donde todos los dispositivos se atienden secuencialmente, entregando la prioridad a todos, y el segundo está con **prioridad escalonada**, en donde la identificación de dispositivos parte del primer barrido que realiza el gestor para reconocerlos.

El segundo tipo de gestión se desarrolla por interrupciones. Aquí se debe reconocer que es una interrupción de ***hardware***, la cual es una señal proveniente de un ***hardware*** de I/O para informar al procesador una actividad desde esta etapa. En el momento que se activa la interrupción, el administrador de dispositivos realiza la acción correspondiente hasta que la termine. Este proceso conecta directamente los periféricos al procesador por medio del bus de datos.

Por último, se presenta una gestión híbrida, en donde se combinan los dos tipos de controles vistos anteriormente, realizando una gestión por interrupción; pero cuando el sistema se satura, el gestor de dispositivos habilita la opción de muestreo escalonado para atender a todos los dispositivos.

Instrucción

Para complementar la información desarrollada acerca de la gestión de dispositivos de entrada y salida se recomienda realizar la **lectura complementaria** "Aplicación de los Sistemas Operativos en Tiempo Real y los Protocolos de Comunicación al Diseño de un Sistema de Adquisición de Datos Modular de Bajo Coste y Bajo Consumo", en la cual se realiza un acople interesante de dos temáticas necesarias para el desarrollo del programa de Ingeniería de Sistemas, en este caso el hardware desde los dispositivos para los equipos de cómputo y su conexión en RED y como fue diseñado un sistema operativo que les permitiera reducir costos para el sostenimiento del sistema.

Además, como ejercicio de apropiación de la temática trabajada en el bloque anterior, desarrolle la **actividad de repaso 2** .

Para finalizar lo invitamos a realizar la actividad evaluativa del eje 2.

BIBLIOGRAFÍA

Bibliografía

Bellido Quintero, E. (2013). Manual de Instalación y Configuración de Sistemas operativos (MF0219_2). Madrid: CEP S.L.

Castillo, J. (2017). SoloCiencia.com. Obtenido de SoloCiencia.com: http://www.solociencia.com/informatica/computador-historia-historia.htm

Muñoz Lopez, F. J. (2012). Sistemas Operativos Monopuestos. España: Mc Graw Hill.

San Marcos

www.usanmarcos.ac.cr

San José, Costa Rica