UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE INGENIERÍA

MONITOREO Y AFINACIÓN DEL RENDIMIENTO DEL SISTEMA OPERATIVO LINUX [MYALIX]

T E S I S

QUE PARA OBTENER EL TITULO DE:
I N G E N I E R O E N C O M P U T A C I Ó N

P R E S E N T A:
MIGUEL ANGEL HERRERA MARTIN DEL CAMPO

DIRECTOR DE TESIS:
DRA. ANA MARÍA VAZQUEZ VARGAS

Ciudad Universitaria, México DF. 2004

Agradecimientos

Agradezco a la Universidad Nacional Autónoma de México, la oportunidad que me dio para conocerla.

Agradezco a la Dra. Ana María Vázquez su infinita paciencia, ya que sin ella este trabajo nunca habría sido realizado.

Agradezco a todos y cada uno de mis maestros, el conocimiento transmitido en las diferentes aulas de la facultad de Ingeniería.

Agradezco el incondicional apoyo de mi familia, de mi amada esposa Elizabeth y de mis adorados hijos Rodrigo Iván y Axel Leonardo.

Agradezco a ti, como te llamen señor, la oportunidad de darnos por alcanzar lo que deseamos.

Índice General

SEGUNDA PARTE: IMPLEMENTACIÓN DEL SISTEMA

El premio más grande que nos da la vida es la oportunidad
de trabajar para lograr lo que realmente deseamos.
Frank D. Roosevelt

1. Introducción

En ésta época en la que la información es piedra angular y que la tecnología avanza día a día, es muy común el hecho de adquirir diferentes tipos de computadoras personales que, en comparación, son mucho más potentes que las que fueron utilizadas para enviar a el hombre a la luna en la década de los sesenta.

A pesar de este gran avance tecnológico, cualquiera de estas computadoras siguen siendo dispositivos electromecánicos con tres características fundamentales, las cuáles son su gran capacidad de almacenamiento, su gran velocidad de procesamiento y su facilidad de programación.

Cada vez la computadora personal, o el ordenador como lo han definido los europeo hispanos, ha tomado posiciones criticas en nuestro entorno social; sencillamente no podemos imaginar realizar transacciones bancarias, hacer reservaciones de cualquier tipo, generar diferentes formas de diseño o simplemente formular una carta o un mensaje, sin contar con el apoyo incondicional de una computadora personal que nos facilite sustancialmente la tarea.

Todas estas máquinas que nos facilitan nuestro que hacer diario, están fundamentalmente compuestas por un cerebro que controla todo el equipo, denominado procesador central, una memoria de almacenamiento, mejor conocida como disco duro, una memoria de trabajo, denominada memoria principal y diferentes dispositivos de entrada/salida que permiten introducir y recibir información que ha sido procesada por la computadora personal.

En nuestros días, para poder adquirir cualquier equipo de cómputo existente en el mercado, bastará con cuantificar cada uno de los componentes que la forman, sin olvidar que la explosiva evolución tecnológica en la que vivimos, harán que nuestro reluciente equipo muy pronto se vuelva obsoleto. Para ese entonces pensaremos en cambiar de equipo sin habernos percatado que tanto provecho podríamos aún obtener de él, ya que no se cuenta con las herramientas adecuadas que nos permita sopesar el como obtener más provecho de nuestro equipo.

El presente trabajo pretende definir y clarificar de la mejor forma posible las principales causas que afectan el rendimiento de uno de los sistemas operativos más utilizados hoy en día en computadoras personales y la forma en la que pueden ser resueltas, para que el desempeño de nuestro equipo sea óptimo.

Pretendemos aportar un sistema que pueda fácilmente detectar conflictos en cualquiera de los diferentes módulos que componen la computadora personal y que al mismo tiempo sugiera acciones a realizar con el firme propósito de erradicarlo o disminuirlo, de ser posible, sin la necesidad de realizar ampliaciones o cambios de alguno de los módulos involucrados.

El plan de trabajo que desarrollaremos en el presente documento lo dividimos en dos partes: antecedentes, que consta de cinco capítulos e implementación del monitor en lenguaje *shell*, que consta de tres. Mismos que presentamos a continuación.

Éste primer capítulo, es una introducción que nos permitirá prever como se realizó el sistema de monitoreo y afinación del rendimiento del sistema operativo Linux.

En el segundo capítulo, recordaremos la forma en que se desarrollaron los diferentes equipos de cómputo con sus respectivos sistemas operativos, haciendo énfasis en el sistema operativo que nos compete. Al capítulo lo denominamos: Historia de los sistemas operativos.

En el tercer capítulo, narraremos la forma de cómo y por que razones realizamos la instalación de los sistemas operativos Windows Xp y Linux en la misma computadora. Al capítulo lo denominamos: Instalación y configuración de Linux.

En el cuarto capítulo, conoceremos lo popular que se ha vuelto el sistema operativo Linux así como las más importantes opciones con las que contamos actualmente. Al capítulo lo denominamos: Distribuciones existentes en el mercado.

En el quinto capítulo, definiremos las principales razones que han permitido hacer tan popular y accesible a este singular sistema operativo. Al capítulo lo denominamos: Razones por las que es tan utilizado hoy en día.

En el sexto capítulo, detallaremos los factores que afectan el rendimiento de cada uno de los recursos existentes en la computadora personal. Al capitulo lo denominamos: Factores que afectan el rendimiento del Sistema Operativo Linux

En el séptimo capítulo, esclareceremos la forma en que realizamos el sistema que permite monitorear y recomendar acciones para afinar el rendimiento del sistema operativo Linux. Al capitulo lo denominamos: Implementación de un Sistema que realice el análisis y afinación del rendimiento de Linux

En el octavo capítulo, encontraremos las conclusiones del presente trabajo. Ahí definiremos si logramos el objetivo buscado que consiste en construir un sistema que nos permita identificar problemas de rendimiento en los diferentes recursos que componen las computadoras personales que tengan instalado el sistema operativo Linux.

En el anexo, recopilamos todos los comandos utilizados en el desarrollo de este sistema de monitoreo, los cuáles se encuentran bien ejemplificados para que puedan ser utilizados como referencia en futuros sistemas que pudieran ser desarrollados.

Al final del trabajo, se encuentra la bibliografía más importante utilizada para el desarrollo de nuestro proyecto.

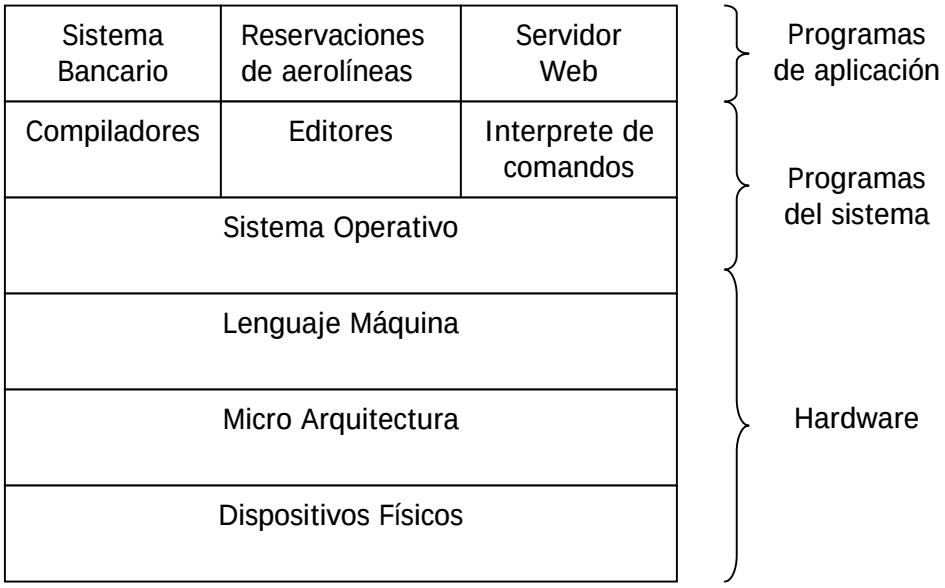
Una vida inactiva es una muerte prematura
Johan W. Goethe

2. Historia de los Sistemas Operativos

Todos los sistemas de cómputo modernos consisten de uno o varios procesadores, memoria principal, discos, impresoras, un teclado, un video, interfaces de red y otros dispositivos de entrada/salida.

El escribir programas que puedan controlar todos éstos componentes y utilizarlos correctamente es un trabajo realmente difícil. Por tal razón, las computadoras están provistas de una capa de software llamada sistema operativo, cuyo trabajo es manejar todos los dispositivos y proveer programas al usuario que le permitan controlar el hardware muy fácilmente.

Las capas que forman el sistema de la computadora son el hardware, los programas del sistema y los programas de aplicación, tal y como se muestra en el siguiente diagrama:



En la parte de inferior del diagrama encontramos al hardware, el cuál a su vez se encuentra compuesto por dos o tres niveles. El nivel más bajo esta formado por los dispositivos físicos, como son los circuitos integrados, cables, fuentes de poder, tubos de rayos catódicos y otros dispositivos más que son popularmente utilizados hoy en día.

En seguida viene el nivel de micro arquitectura, en el cuál los dispositivos físicos son agrupados en forma de unidades funcionales para así facilitar tanto su uso, como control.

El lenguaje máquina tiene entre 50 y 300 instrucciones, la mayoría de ellas son utilizadas para mover datos, realizar cálculos aritméticos y comparar valores. En este nivel los dispositivos de entrada/salida son controlados mediante la carga de valores en registros relacionados con dichos dispositivos.

Para poder ocultar toda esta complejidad, se provee el sistema operativo. Éste consiste de una capa de software que oculta el hardware y proporciona al programador un juego más conveniente de instrucciones con el cuál trabajar más fácilmente.

Sobre el sistema operativo encontramos los demás programas que forman el sistema. Aquí podremos encontrar el intérprete de comandos (shell), compiladores y editores. Es de suma importancia resaltar que estos programas no son realmente parte del sistema operativo.

El sistema operativo es generalmente la parte de software que se ejecuta ya sea en modo *kernel* o supervisor, mientras que los compiladores y los editores se ejecutan en modo de usuario.

Finalmente en la parte superior del diagrama encontramos los programas de aplicación, los cuáles son comprados o realizados por los mismos usuarios para resolver problemas específicos.

Los sistemas operativos han evolucionado a través de los años, los cuáles revisaremos mediante sus generaciones.

Recordemos que la primera computadora digital fue diseñada por el matemático inglés Charles Babbage (1792-1871). Aunque éste matemático inglés invirtió gran parte de su vida y fortuna tratando de construir su *máquina analítica*, nunca pudo hacerla trabajar propiamente ya que era completamente mecánica y la tecnología de sus días no podía producir las ruedas, engranes y dientes de engrane con la precisión requerida. Resulta obvio mencionar que la *máquina analítica* no contaba con un sistema operativo.

Babbage pudo comprender que su máquina analítica requería de una pieza de software, para lo que contrato a una joven mujer llamada Ada Lovelance, quien era hija del afanado poeta inglés Lord Byron, para que la desarrollara; convirtiéndose así en el primer programador (o deberíamos decir, en la primera programadora) de la historia de la computación.

Primera Generación [1945- 1955]
Bulbos y plugboards

Después de los esfuerzos sin éxito de Charles Babbage, poco se realizó en la construcción de computadoras digitales hasta la segunda guerra mundial. Alrededor de los años cuarenta, Howard Aiken de Harvard, John von Neumann de Princeton, J. Presper Eckert y William Mauchley de la Universidad de Pennnsylvania, Konrad Zuse en Alemania, entre otros, consiguieron crear las primeras máquinas calculadoras.

Estas calculadoras utilizaron principalmente *relays,* los cuales resultaron ser muy lentos, motivo por el cuál fueron reemplazados posteriormente por bulbos, mismos que tenían un tamaño considerable.

Estás máquinas eran enormes, requerían de cuartos enteros para almacenar los miles de bulbos que las componían. Al parecer, se requerían de al menos diez mil de ellos.

Curiosamente el termino *debugger* data desde entonces, ya que diferentes tipos de "bichos" se alojaban entre los bulbos, haciendo que no se tuviera un contacto adecuado, por lo que era necesario detener la máquina y limpiar los bulbos de los muy diferentes inquilinos que buscaban el calor generado, a los cuáles se les denominó *bugs*. El término se sigue conservando hasta la fecha, aunque ya no se eliminan insectos, sino errores de programación o lógica.

En aquellos días un simple grupo de personas diseñaba, construía, programaba, operaba y mantenía cada máquina. Toda la programación era realizada en lenguaje máquina, los lenguajes de programación se desconocían y no se tenía aún idea de los sistemas operativos.

Al inicio de los años cincuenta aparecieron las tarjetas perforadas, dando un cambio a la rutina creada. Ahora era posible escribir programas en tarjetas y leerlas desde la computadora, con lo que se ahorraba tiempo. Curiosamente la clasificadora de tarjetas (el antecesor del comando *sort*) era realmente enorme y por ende muy pesada.

Segunda Generación [1955- 1965]
Transistores y Sistemas Batch

La creación del transistor a mediados de los años cincuenta dio un impulso inimaginable a la era de la computación. Por primera vez hubo una clara separación entre diseñadores, constructores, operadores, programadores y personal de mantenimiento, dando así origen a la especialización de cada una de las diferentes áreas que en conjunto dan sentido a la revolucionaria era de la computación.

Estas máquinas, ahora llamadas *mainfraimes,* estaban encerradas en habitaciones especiales con aire acondicionado, con un equipo dedicado a operarlas adecuadamente. Solamente las grandes corporaciones, algunas agencias de gobierno o ciertas universidades podían afrontar el costo multimillonario.

Mucho del tiempo de la computadora era mal gastado por los operadores que caminaban alrededor de la habitación en la que se encontraba. Debido al alto costo del equipo, se buscaron formas que permitieran reducir esta perdida de tiempo y, por su puesto, dinero.

La solución encontraba fue la generación de sistemas *batch*. La idea era concentrar trabajos que serían procesados por la computadora principal desde cinta magnética, misma que era creada por otra computadora más pequeña y, por supuesto, menos costosa.

Para poder leer la cinta magnética, el operador cargaba un programa especial (el antecesor de los actuales sistemas operativos) que leía el primer trabajo de la cinta y lo ejecutaba. El resultado de la ejecución era escrito en una segunda cinta, en lugar de imprimirlo. Al terminar con el primer trabajo alojado en la cinta magnética, el sistema operativo automáticamente leía el siguiente trabajo y lo ejecutaba hasta terminar con todos los trabajos almacenados en la cinta.

El operador entonces cambiaba tanto la cinta de entrada como la de salida para seguir procesando trabajos batch, y colocaba la cinta de salida en la computadora pequeña para así poder imprimir los resultados obtenidos. Esta pequeña máquina no estaba conectada a la computadora principal. La computadora principal era la IBM 7094 y la más pequeña era la IBM 1401.

Los usos principales de los recursos de cómputo fueron enfocados en el ámbito científico y cálculos ingenie riles, tales como la solución de ecuaciones diferenciales. Los programas eran realizados tanto en fortran como en lenguaje ensamblador.

Los sistemas operativos típicos eran FMS (Fortran Monitor System) e IBSYS, que era el sistema operativo de IBM.

Tercera Generación [1965 - 1980]
Circuitos Integrados y multiprogramación

A principios de los años sesenta, la mayoría de las computadoras tenían dos distintas y totalmente incompatibles líneas de productos. De un lado se tenían las computadoras que estaban orientadas al cálculo numérico resolviendo problemas de ingeniería, y en el otro lado se tenían las computadoras utilizadas por bancos y compañías de seguros que hacían uso de procesos batch y cintas magnéticas. Ambos equipos eran modelos de IBM, la 7094 y la 1401.

Desarrollar y mantener dos diferentes líneas de producto era una propuesta muy costosa para los pocos fabricantes, además, muchos nuevos clientes requerían de equipos más potentes que pudieran ejecutar sus ya probados procesos en forma más rápida.

IBM, considerado por mucho tiempo el monstruo azul, intentó resolver estos problemas desarrollando e introduciendo al mercado el sistema 360 que era más potente que la IBM 7094 y realizaba las operaciones propias de la IBM 1401, las cuáles solamente diferían en capacidad y precio. De esta forma una sola familia de computadoras podría satisfacer las necesidades de todos sus posibles clientes.

En los años siguientes IBM introdujo otros modelos con tecnología más moderna, conocidos como la 370, 4300, 3080 y la serie 3090. Aunque la IBM 360 fue la primer línea de computadoras en utilizar circuitos integrados, teniendo un éxito sin precedente.

La gran idea de tener una sola familia de computadoras, era al mismo tiempo su mayor debilidad, ya que la intención era tener en el sistema operativo OS/360 toda la programación necesaria para todos sus modelos utilizados en muy diferentes usos.

El resultado fue un enorme y muy complejo sistema operativo, con miles de líneas escritas en lenguaje ensamblador por cientos de programadores, el cuál seguramente era tres veces más grande que el anterior FMS.

Uno de los diseñadores del sistema operativo OS/360, Fred Brooks, escribió en el año 1996 un libro describiendo su experiencia en su desarrollo. Un hecho inevitable era que cuando se quería corregir alguna deficiencia, generalmente se generaban nuevas, haciendo así eterno el proceso de perfeccionamiento del sistema operativo.

A pesar de todos los problemas encontrados, el sistema operativo OS/360 pudo resolverlos bastante bien. Incluso ayudo a la generación de nuevas técnicas de desarrollo, de las cuales seguramente la más importante fue el concepto de multiprogramación.

Este concepto realiza la partición de la memoria principal en diversos fragmentos, colocando un proceso diferente en cada partición. Así cuando un proceso esté esperando la escritura en disco, otro proceso podría hacer uso del procesador central.

Otro rasgo de los sistemas operativos de la tercera generación fue la habilidad de poder leer trabajos directamente de las tarjetas perforadas al disco.

Así en cuanto el proceso terminara, el sistema operativo podría cargar un nuevo trabajo del disco a alguna de las particiones de memoria para ejecutarlo. Esta técnica fue conocida como *spool*, ya que tomó sus siglas de *Simultaneous Peripheral Operation On Line,* la cuál también fue utilizada para los dispositivos de salida. Con ésta nueva técnica el equipo IBM 1401 y la mayoría de las unidades de cintas magnéticas, cayeron en desuso.

El deseo de tener un mejor tiempo de respuesta, pavimento el camino para el concepto de tiempo compartido (*timesharing*) en donde cada usuario contaría con una terminal conectada directamente a la computadora, permitiéndoles de esta forma optimizar el tiempo de programación y la depuración de errores.

Recordemos que inicialmente los programadores debían “capturar” en tarjetas perforadas todas las líneas de código, cada tarjeta representaba solamente una línea del programa, por lo que era común ver cajas llenas de éstas tarjetas para completar el programa, dependiendo claro está, de la complejidad del mismo. Estos programadores debían esperar a que fueran leídas todas las tarjetas por la computadora central para poder revisar sus errores varias horas después y repetir nuevamente el proceso hasta erradicar todos los errores cometidos.

El primer sistema serio de tiempo compartido, CTSS (*Compatible Time Sharing System*), fue desarrollado en el MIT. Sin embargo, no se volvió popular hasta que la necesaria protección del hardware se extendió durante la tercera generación.

Después del éxito del sistema CTSS, el MIT, los laboratorios Bell y la compañía General Electric decidió desarrollar una utilería que soportaría cientos de usuarios simultáneos. Los diseñadores del sistema, conocido como MULTICS (*MULTiplexed Information and Computing Service*), concebía una gran computadora que soportara gran cantidad de

usuarios simultáneos. MULTICS introdujo una gran cantidad de ideas seminales en la literatura de la computación.

Otro gran desarrollo durante la tercera generación fue el fenomenal crecimiento de las mini computadoras, iniciando con la DEC PDP-1 en 1961, la cuál tenía 4 kilobytes de memoria de trabajo y costaba menos del 5 por ciento de la IBM 7094. Este hecho hizo que se vendiera como pan caliente.

Este equipo fue seguido por la serie de la familia PDP, que a diferencia de IBM eran todas incompatibles entre sí, culminando con la famosa PDP-11.

Uno de los científicos de los laboratorios Bell que había participado en el proyecto MULTICS, Ken Thompson, empezó a trabajar con una de éstas mini computadoras, la DEC PDP-7; éste trabajo permitiría posteriormente el desarrollo del sistema operativo Unix, el cuál se volvió popular en el mundo académico, agencias de gobierno y en muchas otras.

Gracias a que el código fuente del sistema operativo Unix podía ser conseguido fácilmente, el desarrollo surgió por doquier. Dos fueron las versiones rectoras, el System V de AT&T y BSD (*Berkeley Software Distribution*) de la Universidad de Berkeley en California, las cuales tienen ligeras diferencias entre si.

En 1987 Andrew Tanenbaum desarrolló un Unix recortado con fines académicos al cuál nombró Minix, precisamente por ser un Unix miniatura. La funcionalidad de Minix era muy similar a Unix.

Minix podía ser conseguido gratuitamente, de tal forma el estudiante finlandés Linus Torvalds y un gran grupo de colaboradores pudieron realizar el sistema operativo Linux.

Cuarta Generación (1980 hasta nuestros días)
Computadoras Personales

Con el gran desarrollo de los circuitos integrados, pudieron integrarse cientos y hasta miles de transistores en un centímetro cuadrado de silicón, la era de las computadoras personales había comenzado.

Desde el punto de vista de su arquitectura, las computadoras personales (inicialmente conocidas como micro computadoras) no eran tan diferentes de las mini computadoras de la clase PDP-11, pero en términos de precio ciertamente eran muy diferentes.

Las mini computadoras hicieron posible para muchas compañías o universidades tener su propia computadora, el micro procesador hizo posible pensar en utilizar computadoras personales.

En 1974 Intel introdujo el procesador 8080, el primer CPU de 8 bits de propósito general. Intel asignó a uno de sus consultores, Gary Kildall, el desarrollo del sistema operativo que lo controlaría. Kildall escribió un sistema operativo basado en discos llamado CP/M

(*Control Program for Microcomputers*). Kildall entonces creo la compañía Digital Research para futuros desarrollos y ventas.

En 1977 la compañía Digital Research reescribió el sistema operativo CP/M para que pudiera ser utilizado por diversas micro computadoras que utilizaban, ya sea el procesador 8080, el Z80 (de zilog) u otros procesadores.

A principio de los años ochenta, IBM diseño su IBM PC e inicio la búsqueda de software que pudiera ser ejecutado en esta nueva computadora. Gente de IBM contacto a Bill Gates para autorizar el uso de su interprete BASIC, incluso le preguntaron si conocía de algún sistema operativo que pudiera ser ejecutado en su IBM PC.

Bill Gates sugirió que se contactara a la compañía Digital Research, que en ese momento era la compañía dominante en la creación de sistemas operativos. Increíblemente Kildall rehusó reunirse con IBM y envío a un representante; este hecho fue registrado como la peor decisión en la historia de la computación.

Consecuentemente IBM regresó con Bill Gates para solicitarle la generación de un sistema operativo. Gates contactó una compañía manufacturera local, Seattle Computer Products, quienes tenían convenientemente un sistema operativo llamado DOS (*Disk Operating System*), el cuál compró por 50,000 USD.

Así Bill Gates pudo ofrecer a IBM un paquete formado por DOS y el intérprete de BASIC para su nueva computadora, el cuál aceptaron.

Cuando IBM requirió de ciertas modificaciones, Gates contrato a quien desarrollo el *Disk Operating System*, Tim Paterson. La nueva versión fue llamada MS-DOS (*Microsoft Disk Operating System*) y rápidamente empezó a dominar el mercado de las computadoras personales.

En 1983 apareció la IBM PC/AT con un procesador Intel 80286, MS-DOS fue colocado firmemente y el CP/M estaba perdiendo toda su fuerza. El sistema operativo MS-DOS fue después ampliamente utilizado en los equipos 80386 y 80486, tomando un lugar que ya nunca perdería.

A pesar de que la primera versión del sistema operativo MS-DOS era rudimentaria, las siguientes versiones incluyeron rasgos propios de Unix que la hicieron un poco más robusto. Microsoft estaba pendiente del desarrollo de Unix, incluso vendió computadoras con su versión llamada Xenix durante los primeros años de la compañía.

CP/M, MS-DOS y otros sistemas operativos de las primeras microcomputadoras debían utilizar el teclado para introducir los comandos al sistema. Todo este ambiente cambio gracias a la investigación realizada por Doug Engelbart en el *Stanford Research Institute*.

Engelbart inventó el ambiente denominado GUI (*Graphical User Interface*), que permitía el uso del sistema sin necesidad de teclear comando alguno. Lo complemento con ventanas, iconos, menús y el insustituible mouse.

Steve Jobs incorporó este ambiente en su computadora Apple, misma que construyó en su propia cochera. Una segunda versión perfeccionada de su equipo, Apple Macintosh, tuvo un gran éxito en el mercado, debido al novedoso ambiente de desarrollo en el que se fundamentaba, ya que resultaba ser muy amigable para el usuario final.

Las pantallas verde ó ámbar tipo carácter denominadas CUI (*Character User Interface*), estaban por caer completamente en el olvido.

Cuando Microsoft decidió construir el sucesor del MS-DOS, se influenció fuertemente por el éxito obtenido por Macintosh. Entonces produjeron un sistema gráfico al cuál bautizaron como Windows, el cuál inicialmente era ejecutado sobre el original MS-DOS.

Por un lapso de diez años, de 1985 a 1995, Windows era solamente un ambiente gráfico sobre el sistema MS-DOS. En ese año 1995, se liberó una nueva versión de Windows, denominada Windows 95 que incorporaba muchos rasgos del sistema operativo, utilizando la capa de MS-DOS solamente para inicializar el equipo y para ejecutar programas realizados en el anterior MS-DOS.

En 1998, una versión ligeramente modificada del Windows 95, fue liberada con el nombre de Windows 98. A pesar de las mejoras realizadas, ambos Windows seguían teniendo una gran cantidad de lenguaje ensamblador. No olvidemos que el procesador utilizado era Intel a 16 bits.

Después fue liberado Windows NT (NT eran las siglas de *New Technology*), el cuál era compatible con Windows 95 hasta cierto nivel y que estaba completamente reescrito. Era un sistema de 32 bits. El diseñador fue David Cutler, quien fue también uno de los diseñadores del sistema operativo VMS de VAX y que incluyó en NT algunas de sus funcionalidades.

Microsoft esperaba que la primera versión de Windows NT erradicara las anteriores versiones de Windows ya que era un sistema operativo muy superior, pero desafortunadamente no sucedió así. Fue hasta la versión 4.0 de Windows NT cuando se pudo acaparar un mayor número de usuarios finales.

La versión 5 de Windows NT fue lanzada al mercado como Windows 2000 a principios de 1999, con la idea de unificar a los usuarios de Windows 98 y Windows NT 4.0, lo cuál tampoco funcionó como se había planeado; entonces se introdujo al mercado otra versión de Windows 98 a la cuál se le llamo Windows Me (*Millennium edition*).

El otro gran competidor en el mundo de las computadoras personales es Unix y sus derivados. Unix es un sistema operativo más confiable y estable, por lo que es ampliamente utilizado en estaciones de trabajo y servidores de red. En las computadoras construidas con procesadores Pentium, Linux se ha convertido en una excelente alternativa para los estudiantes de informática y para la creciente demanda.

A pesar de que la mayoría de los usuarios Unix prefieren el ambiente carácter al gráfico, actualmente casi todos los ambiente Unix soportan un excelente ambiente gráfico denominado X Windows, el cuál fue desarrollado completamente por el MIT.

Este sistema incluye el movimiento básico de ventanas, permitiendo a los usuarios crear, borrar, mover y adecuar el tamaño de las ventanas utilizando solamente el mouse.

Generalmente un ambiente completo GUI, como lo es Motif, está disponible para ser ejecutado sobre X Windows, dando a Unix una apariencia muy similar a la creada por Microsoft Windows o Macintosh, para aquellos usuarios que la prefieran.

Un interesante desarrollo tomó lugar a mediados de la década de los ochenta y fue el crecimiento de redes de computadoras personales, las cuáles hacían uso de sistemas operativos de red y sistemas operativos distribuidos. Este nuevo auge creo la bien conocida era de Internet que sigue, y seguramente seguirá, vigente por mucho tiempo.

En los sistemas operativos de red, los usuarios están concientes de la existencia de múltiples computadoras conectadas entre sí, por lo que se tiene la posibilidad de realizar transferencia de archivos entre computadoras remotas. Cada computadora ejecuta su propio sistema operativo localmente y cuenta con sus propios usuarios (o usuario).

Los sistemas operativos de red no son muy diferentes de los sistemas operativos que controlan un solo procesador (conocidos también como mono procesador), ya que requieren además de un controlador de red y de ciertos programas que lo puedan manipular. Estas pequeñas adiciones no cambian la estructura esencial del sistema operativo.

En cambio, en un sistema operativo distribuido necesita tener el control de múltiples procesadores. Con ello el sistema operativo será el encargado de tener el control de la ejecución de los procesos entre los diferentes procesadores identificados, almacenar adecuadamente la información de cada uno de los usuarios, controlar el acceso de los usuarios entre los diferentes equipos, hacer uso eficiente de los manejadores de bases de datos instalados, entre muchas otras responsabilidades.

Los sistemas distribuidos, o al menos los que se jacten de serlo, permiten que una aplicación sea ejecutada en diferentes procesadores al mismo tiempo, lo que requiere una lógica mucho más compleja para el control y distribución de carga de trabajo entre los procesadores existentes.

Como podemos claramente apreciar, la evolución de los sistemas operativos ha sido realmente impresionante a lo largo de los casi sesenta años de historia. Abordemos ahora el sistema operativo que nos concierne y en el cuál está fundamentado el presente trabajo.

Historia de Linux

El nacimiento del sistema operativo Unix en el año 1969, marcó el inicio de toda una era de la informática. Unix fue desarrollado por Ken Thompson y Dennis Ritchie [Linux guía de instalación y administración] dentro del grupo de investigación de AT&T.

La distribución gratuita del sistema operativo, junto con el código fuente, contribuyó decisivamente a su rápida difusión y mejora, consiguiendo con ello que Unix se convirtiera en uno de los sistemas operativos más utilizados en el mundo.

La facilidad de acceder al código fuente de Unix fue, en aquel momento, la forma primordial del desarrollo de éste singular sistema operativo, ya que uno de los pilares rectores fijados para su distribución impedía su comercialización.

Desafortunadamente esta idea se desvaneció del mundo de los fabricantes de software cuando iniciaron su comercialización. A partir de ese momento, se negó el acceso a los programas fuente.

Al contar con los programas fuentes y el "*know how*" del sistema operativo, el conocido prolífico autor holandés Andrew S. Tanenbaum desarrolló el sistema operativo Minix; éste sistema operativo pretendía ser una implementación básica y reducida de Unix, algo así como un Unix en miniatura o de bolsillo, tal y como su nombre lo indicaba.

Tanenbaum no estaba guiado por intereses comerciales, sino que su fin era primordialmente académica y docente; pretendía que sus alumnos pudieran aproximarse a Unix y practicar el manejo del mismo en sus computadoras personales sin tener que afrontar el pago de una costosa licencia.

Así Tanenbaum desarrolló en su totalidad el código fuente de Minix (en lenguaje C y ensamblador) y lo incluyó como ejemplo de implementación y diseño de un sistema operativo en uno de sus libros: *Operating Systems: Design and Implementation,* publicado en 1988 [Linux guía de instalación y administración].

De esta forma, muchos estudiosos de los sistemas operativos tuvieron libre acceso a un código que les permitiría conocer y sobretodo adentrarse en el mundo Unix.

Aunque se vendieron los derechos de Minix, éste sembró la semilla de la que germinaría Linux, ya que la primera versión de este sistema operativo pretendía simplemente ser una mejora de Minix.

En muchas ocasiones se ha definido a Linux como "un Unix para PC", los cuál no es totalmente correcto, ni define plenamente al sistema operativo, pero sí permite formarse una idea aproximada sobre cuál es su origen y sobre todo cuáles son sus características más importantes; mismas que le han dado un lugar preferencial entre sus múltiples usuarios y que seguramente no perderá.

El creador de Linux

El creador de Linux, Linus Torvals, nació en Helsinki, Finlandia el 29 de julio de 1970. El era estudiante de informática en la Universidad de Helsinki a principios de los noventa y como muchos estudiantes de su época, pasaba gran parte de su tiempo practicando con Minix.

A pesar de su innegable talento, nadie podía imaginar en ése momento que una década después sería conocido y respetado en el mundo entero como una de las personalidades clave en el mundo de la informática.

En 1991, Linus se encontraba trabajando en un proyecto propio para mejorar Minix, ya que le encontraba ciertas deficiencias, por lo que se enfocó en desarrollar un sistema similar al Minix que funcionara mejor. Inició desarrollando este proyecto en una computadora personal 386 utilizando ensamblador, pero pronto pudo percatarse que realizarlo en lenguaje C aceleraría su desarrollo.

De esta forma, Linus Torvalds creó lo que puede considerarse la primera versión de Linux (nombre que surgió de la contracción de Linus y Unix), misma que posteriormente el autor denominaría como versión 0.01; ya que por convenio, la primera versión de un programa debe ser completamente operativa y funcional.

Esto permite apreciar que la versión 0.01 de Linux no era más que el juguete de éste programador. Esta primera versión sólo contenía en realidad un *kernel* muy rudimentario, y para poder realizar cualquier operación era imprescindible que la computadora también tuviera instalado Minix.

La versión 0.01 nunca fue difundida. Linus Torvalds continuó modificando el nuevo sistema operativo, y no lo hizo público hasta que consiguió ejecutar ciertos programas del proyecto GNU (*Graphical Not Unix*), como el bash (*Bourne Again Shell*) y el gcc (*GNU C Compiler*). Esta nueva versión de Linux sería denominada como 0.02, y sería la primera versión de Linux que su autor considerará mínimamente utilizable, a pesar de sus muchas limitaciones, y de suficiente nivel como para ser publicada.

La publicación de la versión 0.02 de Linux tuvo lugar el 5 de Octubre de 1991, en un foro de debate sobre Minix. Linus distribuyó de forma gratuita en este foro el código fuente de Lnux, invitando a la participación activa en el proyecto a cualquier persona con interés en el tema y, por su puesto, con los suficientes conocimientos de programación como para examinar el código y aportar mejoras o añadidos al recién creado sistema operativo, así como también para detectar posibles errores o aportar nuevas ideas.

Esta llamada a la colaboración tuvo un eco tal que, a partir de ese punto, se puede decir que Linux dejó de ser obra exclusiva de Linus Torvalds para pasar a ser el fruto del trabajo colectivo de un equipo formado por un gran número de colaboradores voluntarios.

El trabajo de todos

Es un hecho innegable que el desarrollo de Linux no hubiera sido posible sin el trabajo y el tiempo dedicados al sistema por parte de los miles de programadores que, antes o después, aceptaron la invitación para colaborar en el proyecto Linux a través de la red de redes (Internet). Todos ellos, en mayor o menor medida, pero siempre de forma altruista, han aportado al sistema operativo sus conocimientos, sus ideas y grandes dosis de entusiasmo. El trabajo en paralelo de toda esta comunidad permitió completar, depurar y mejorar las distintas versiones de Linux a un ritmo imposible de conseguir para un solo programador.

El crecimiento del código fuente fue de tal magnitud, que se estima que Linus Torvalds no ha escrito más allá del 5 por ciento del código total de Linux (aunque esta estimación no es sencilla, dadas las múltiples versiones y correcciones del sistema operativo). Este hecho no significa que Linus Torvalds perdiera protagonismo en el proceso, sino al contrario, Linus siguió asumiendo el papel principal en la dirección y coordinación del proyecto.

El poder desarrollar un proyecto en estas circunstancias exige no solamente voluntad e ideas, sino también que las distintas aportaciones al sistema operativo se realicen en forma ordenada y sobretodo coordinada. Es posible que las mejoras realizadas por un colaborador no sean realmente acertadas y necesarias.

También es posible que varios colaboradores trabajen simultáneamente en solucionar un mismo problema, solapándose así en su trabajo. Para evitar estos problemas, un grupo de colaboradores (entre los cuáles se encontraba incluido Linus Torvalds), considerados de suficiente nivel y prestigio, asumieron la tarea de seleccionar las aportaciones realmente relevantes y de coordinar el trabajo de los distintos colaboradores voluntarios.

Gracias a este esfuerzo común, Linux evolucionó de forma exponencial, sucediéndose rápidamente las versiones: 0.03, 0.10, 0.11 y 0.12. En marzo de 1992 se produjo un avance considerable al realizarse la versión 0.95. En la misma época comenzaron a desarrollarse aplicaciones diseñadas específicamente para Linux. El sistema ya podía usarse con el entorno gráfico X-Window, facilitando así el manejo y empleo del sistema operativo desde el punto de vista del usuario final.

Poco tiempo después, llegó la versión 0.96; lo cuál indicaba que Linux se aproximaba a una versión verdaderamente completa, o al menos, así lo consideraba la comunidad de programadores implicados en el proyecto.

Sin embargo, esto no sucedería hasta pasado cierto tiempo, ya que la primera versión completa y sin errores de Linux sería presentada hasta octubre de 1994. En aquel momento, algunos colaboradores enfocaron sus esfuerzos hacia la creación de documentación de calidad y que también fuera gratuita, lo cual propició inevitablemente la proliferación de nuevos usuarios del recién creado sistema operativo. A su vez, las necesidades de estos nuevos usuarios renovaron el interés en el desarrollo de aplicaciones finales y nuevos controladores de hardware para Linux.

La repentina y rápida evolución de Linux no terminó con la publicación de la versión 1.0, sino que continúa incluso hoy en día, siempre mediante el trabajo en equipo, con la firma intención de perfeccionar cada vez más el sistema operativo y de disponer de mejores y más diversas aplicaciones.

La versión 2.0 tardaría en llegar dos años más, la cual fue anunciada el 9 de junio de 1996, e introducía considerables innovaciones respecto a sus versiones predecesoras. Curiosamente la computadora donde se escribió la nueva versión, funcionaba con la versión 2.2.14 del núcleo de Linux. Como podemos apreciar, la evolución del sistema ha sido y sigue siendo constante.

A lo largo de esta evolución, la cantidad de colaboradores del proyecto Linux ha crecido de forma considerable. Paralelamente, se ha producido un espectacular incremento en el número de usuarios de este sistema operativo, sobre todo tras la publicación de las versiones completas. Linux que en un principio parecía reservado a iniciados en informática o al ámbito universitario, ha acabado por llegar al gran público, ya sea por Internet o bien mediante las distribuciones comerciales.

Este éxito en la evolución y difusión de Linux no hubiera sido posible sin la existencia de dos fenómenos sociales de nuestro tiempo: el auge de la filosofía del software libre (conocida también como software libre distribución o código abierto) y la popularidad alcanzada a nivel mundial de Internet.

La filosofía del software libre

Linux fue desarrollado sin ningún ánimo de lucro, ofreciendo siempre el código fuente del programa en forma gratuita. El principal beneficio obtenido por Linus fueron las mejoras realizadas en el sistema por todos los colaboradores del proyecto, que también desempeñaban esta labor sin recibir ninguna compensación económica directa. El código fuente desarrollado por estos colaboradores para mejorar el sistema también se distribuye abiertamente. Esta forma de actuación se ajusta plenamente a los postulados de la filosofía del software libre, a la que Linus era y sigue siendo afín.

El software libre, que como hemos mencionado es un software de libre distribución o software de código abierto, es un concepto que se ha arraigado con fuerza en la mentalidad de numerosos programadores, originando un movimiento social de consecuencias impredecibles.

Sin olvidar que se trata de un fenómeno complejo y rico en matices, se tratará de exponer sus planteamientos esenciales. Es de suma importancia el dejar claro que la denominación inglesa de esta idea *Free Software* ha originado ciertas confusiones y malas interpretaciones a causa de la ambigüedad de la palabra *free,* que en inglés significa tanto libre como gratis. Realmente, un programa se considera software libre para un usuario si éste, además de poder ejecutarlo, puede modificarlo para adaptarlo a sus necesidades (lo que implica que el usuario tenga libre acceso al código fuente del programa) y puede distribuir copias tanto de la versión original del programa como de las versiones

modificadas realizadas por el usuario. Esta distribución puede realizarse en forma gratuita o no, pero siempre con el objetivo de que el programa acabe mejorado y evolucionado mediante las distintas modificaciones realizadas.

Evidentemente, estos planteamientos tienen una gran resistencia por la mentalidad clásica de comercialización del software, en la que siempre se cobra una cantidad al usuario por el uso del programa y nunca se le permite el acceso al código fuente del mismo, considerándose además delito de piratería informática la realización y distribución de copias del programa.

Siguiendo esta revolucionaria filosofía de forma un tanto radical, cuando Linus hizo público el código de las primeras versiones de Linux, lo realizó gratuitamente y según los términos de una licencia restrictiva que impedía cualquier tipo de distribución comercial del programa. A principios de 1992, Linus suavizó su postura y puso el sistema bajo los términos de una licencia ligeramente menos restrictiva conocida como licencia GPL (*General Public License*) para programas.

Esta licencia permite la distribución comercial del programa, pero siempre de manera que éste haya sido o no modificado, debe seguir sujeto a las normas del software libre. Es decir, cuando el autor de un programa los distribuye bajo los términos de la licencia GPL, permite que el programa sea ejecutado, modificado, copiado y distribuido (incluyendo su venta), pero no permite que el programa original o las versiones modificadas del mismo dejen de ser software libre. Conviene no olvidar que existe otra licencia GPL para bibliotecas, llamada LGPL (*Library General Public License*) que establece normas ligeramente distintas a la licencia de programas.

Con esto queda claro que las distribuciones comerciales de Linux, que actualmente gozan de cierto éxito, no contradicen las ideas propugnadas por el software libre. Además, la cantidad que hay que pagar actualmente para adquirirlas es relativamente modesta: el distribuidor realmente cobra el soporte físico de los datos y el trabajo de recopilación de software. Y, por supuesto, el comprador obtendrá software libre, teniendo acceso al código y pudiéndolo modificar, copiar, etc.

La licencia GPL es el reflejo de la idea conocida como *copyleft*. A diferencia del *copyright* tradicional, el *copyleft* permite al autor del programa ser reconocido como tal, pero no para obtener beneficio o imponer restricciones que impidan que el programa sea modificado, copiado o distribuido, sino para impedir que otra persona se apropie del programa (total o parcialmente) e imponga aquellas restricciones sobre su uso que el mismo autor no ha impuesto.

Puesto que la licencia GPL establece que las versiones modificadas de los programas de software libre deben seguir siendo software libre, la comunidad de programadores que colabora en un proyecto bajo licencia GPL se beneficia inmediatamente de cualquier mejora que se realice en el programa, lo que favorece su evolución y depuración. El mejor ejemplo de que este planteamiento no es solamente una teoría es sin duda el caso del propio Linux.

La licencia GPL (para programas y bibliotecas) son unos de los muchos frutos del proyecto GNU. El objetivo concreto de este proyecto es desarrollar un sistema operativo completo, similar a Unix, pero sin ser Unix y que sería distribuido como un software libre.

No se trata del único proyecto con este objetivo: en la Universidad de Berkeley se ha venido desarrollando, paralelamente al sistema GNU y con idénticos planteamientos, el sistema operativo BSD [*Berkeley Software Distribution*]. El proyecto GNU está promovido desde 1984 por Richard Stallmann, quien es otro de los promotores del movimiento del software libre (e incluso más radical que Linus en sus planteamientos).

No en vano, Stallmann es uno de los fundadores de la FSF (*Free Software Fundation*), que es una fundación destinada a promover el desarrollo y uso de software libre y recabar ayudas (incluyendo aportaciones económicas y donaciones de material informático) para continuar adelante con el propio proyecto GNU.

El proyecto GNU y Linux están relacionados más allá del plano meramente ideológico. Linus Torvalds comprobó el funcionamiento de sus primeras versiones de Linux compilando y ejecutando determinados programas GNU como el bash o el gcc. En aquella época GNU no había conseguido desarrollar un núcleo estable para su sistema operativo, pero sí una buena cantidad de aplicaciones de gran utilidad para el sistema (como son shells, compiladores, etc.), que habían adquirido cierta popularidad.

Dado que Linux pronto demostró su funcionalidad, la elección de éste como núcleo del sistema GNU era evidente. Desde entonces, se estableció una asociación natural entre Linux y el software desarrollado por el proyecto GNU que ha continuado hasta la fecha.

Obviamente, esta asociación se ha visto favorecida por el hecho de que tanto Linux como los programas GNU son software libre.

Hoy en día, prácticamente la totalidad de los usuarios de Linux emplean también aplicaciones GNU en sus sistemas operativos. De hecho dichos sistemas han sido denominados en ocasiones SISTEMAS GNU/LINUX, para destacar que el software GNU empleado habitualmente, aunque no forme parte del núcleo, sí resulta fundamental para el funcionamiento del sistema global.

Aunque esta denominación ha sido empleada en ciertos círculos, actualmente existe una cierta tendencia a referirse simplemente a SISTEMAS LINUX. Esta tendencia se justifica por el hecho de que no todo el software empleado en sistema Linux es software GNU, pero también por cierto desconocimiento de la realidad del sistema por parte de muchos usuarios, que cotidianamente emplean software GNU sin ser conscientes de ello.

La importante ayuda de internet

A esta altura, y dada su gran popularidad, existen muy pocos aspectos de Internet que no hayan sido explicados, comentados y valorados en la multitud de libros, estudios, artículos y reportajes realizados sobre el tema. Tomando esto en cuenta, el objetivo de este

apartado no es el explicar los fundamentos teóricos de la red de redes, ni analizar su innegable impacto social, sino el de resaltar su importancia en el proceso de la frenética evolución y popularización de Linux.

La importancia de Internet en este proceso es evidente si recordamos que Linux, al menos desde cierto punto de su existencia, ha evolucionado como una obra colectiva. La filosofía de software libre propició este tipo de desarrollo y la licencia GPL ofreció incluso una base legal para el mismo, pero es evidente que nada de esto hubiera servido si no hubiera existido un medio para que todas las personas involucradas, en mayor o menor grado, en el proyecto Linux (comenzando por el propio Linus Torvalds) pudieran compartir sus mejoras, opciones y correcciones al sistema.

Internet, y más concretamente, el correo electrónico y los foros de debate, constituyeron los medios idóneos para la comunicación entre los múltiples colaboradores del proyecto.

Como se menciono con anterioridad, el nacimiento de Linux en la red se produjo en un foro de debate sobre Minix. El interés despertado fue tal que, en poco tiempo, el sistema ya contaba con varios foros de debate propios, en los que tuvieron lugar buena parte de las liberaciones que hicieron evolucionar el sistema operativo. Al mismo tiempo, la red permitía que el código fuente de cualquier contribución a Linux estuviera disponible inmediatamente para ser examinado o probado por cualquier interesado en el tema, o para ser incorporado al sistema una vez aprobada su validez.

Por otra parte, una vez que el sistema estuvo mínimamente completo, comenzaron a aparecer las primeras páginas Web dedicadas a Linux. La proliferación de estas páginas y la popularidad alcanzada por Internet en todo el mundo ha permitido que el número de personas que entran en contacto con Linux mediante la red crezca día a día.

Este hecho, unido a la disponibilidad gratuita del sistema operativo, ha originado que el número de usuarios del sistema aumente hasta alcanzar, hoy en día, las decenas de millones. Así que, Internet no solamente ha sido el medio que ha permitido y sigue permitiendo la construcción y evolución de Linux, sino que también constituye actualmente el principal medio para la difusión del sistema operativo.

No cabe duda de que esta expansión ha contribuido en forma decisiva a afianzar definitivamente el posicionamiento de Linux como un sistema operativo completamente funcional. No importa que los recién llegados a este sistema aporten o no algún tipo de mejora concreta al proyecto global, ya que cualquier usuario que instale Linux en su computadora personal y opere con él ya estará comprobando implícitamente su gran funcionalidad, efectuando así cierto control de calidad del mismo.

Hoy en día, los usuarios de Linux contemplan la red no solamente como una fuente para obtener el código del núcleo del sistema y del software asociado, sino que también como el mejor lugar para buscar una solución a las dudas o problemas planteados por el sistema operativo.

Precisamente uno de los defectos más frecuentes achacados a Linux es la carencia de un servicio técnico oficial que responda a cualquier fallo de funcionamiento (tal y como lo

hacen las grandes compañías de software comercial), pero este hecho pierde importancia si se considera que el soporte brindado espontáneamente, a través de la red de redes, por la comunidad Linux mundial puede ser, muy probablemente, mejor y mucho más rápido que el ofrecido por cualquier gran compañía comercial.

Debemos recordar que aunque las distribuciones comerciales de Linux permiten acceder al código fuente del sistema operativo sin necesidad de obtenerlo a través de la red, también recurren frecuentemente a Internet para así poder ofrecer a sus múltiples usuarios actualizaciones, correcciones y documentación.

Se debe resaltar que Linux fue creado siguiendo las premisas de Unix, mismas que lo definen como un sistema operativo robusto, estable, multiusuario, multitarea, multiplataforma y con gran capacidad para la gestión de redes.

Como las grandes estrellas de la historia de la computación, Linux se ha ganado la confianza de los sectores más influyentes del mundo de la informática, como es el universitario, para comenzar a introducirse con fuerza en el mundo empresarial disputándole el puesto a sistemas operativos sustentados por grandes corporaciones y enormes presupuestos.

En muchas ocasiones es difícil comprender al mundo empresarial, pues los empresarios prefieren adquirir un producto de gran costo antes que uno gratuito, guiados por el criterio no siempre válido en estos tiempos: si es gratuito no puede ser un producto de calidad.

Quizá sea más adecuado decir que el empresario paga por un producto o programa con la clara idea de poder cubrirse la espalda, y de este modo tener la tranquilidad de poder recurrir al fabricante de software en caso de catástrofe, aunque muchas veces esto no haya servido de mucho en circunstancias realmente adversas.

En este momento, los responsables de las área de informática de muchas corporaciones se enfrentan a esta disyuntiva: Linux o Windows. En torno a la posible elección de Linux suelen aparecer las siguiente preguntas: ¿quién solucionará los problemas cuando sea necesario?, es decir, ¿quién proporcionará el soporte técnico?, o lo que es lo mismo, ¿quién está detrás de Linux?

En sentido estricto, Linux tiene el respaldo de miles de programadores que trabajan de modo altruista en dar soporte al producto, pero esto no es todo.

En los últimos años han ido apareciendo distribuidores de Linux que comenzaron haciendo una labor de recopilación y difusión de versiones del sistema operativo junto con las aplicaciones y utilidades más estables que, además, gozaban de cierto prestigio entre los usuarios de Linux.

La novedad consiste en que ahora ofrecen no solamente la distribución de Linux sino que proporcionan también soporte directo al usuario. De esta manera, hoy en día tenemos el sistema operativo Linux bajo denominaciones diversas a modo de apellidos, como son

cada una de las distribuciones que el usuario puede adquirir de modo gratuito en Internet o a cambio de una módica suma.

El reconocimiento de las prestaciones de Linux se reflejan de modo llamativo en determinados sectores, como es el caso de los servidores de Internet, el uso de los cajeros automáticos e incluso en el uso de manejadores de bases de datos tanto de forma local como de forma distribuida.

Las principales versiones de Linux accesibles hoy en día son: RedHat, Mandrake, SUSE, Debian, etc. Todas ellas son Linux, cada una aporta una nota distinta, como el tipo de aplicaciones que instala o, principalmente, las características existentes en el programa de instalación.

La selección de cuál de las diferentes versiones de Linux utilizar en el desarrollo del presente trabajo no fue sencillo, pero un factor determinante en su elección fue el reconocer cuál de ellos es el más utilizado y por ende, el más conocido y respetado en el ambiente Linux, el cuál definitivamente es RedHat.

Más vale una palabra a tiempo que cien a destiempo.
Miguel de Cervantes.

3. Instalación y configuración de Linux

Con la idea de poder desarrollar el presente tema de tesis, decidimos hacer uso de nuestra computadora personal, ya que se podrían realizar las adecuaciones necesarias sin afectar a terceros y se trabajaría a un ritmo sin estar condicionado a la disponibilidad de equipo en alguno de los centros de cómputo, de los muchos existentes en la Universidad Nacional Autónoma de México.

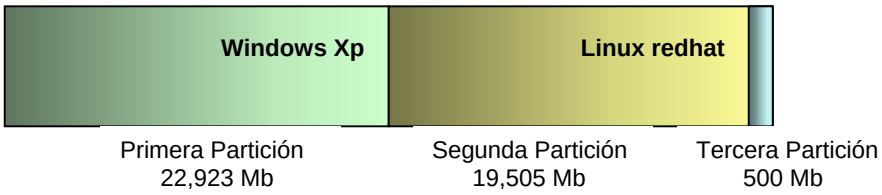
El equipo utilizado es una Compaq Presario 5000 tipo gabinete, con un procesador Pentium III a 750 Mhz, cuenta con un disco duro de 40 Gb, 128 Mb en memoria RAM, tiene lector y grabador de CD, así como modem. Inicialmente el equipo contaba con el sistema operativo Windows 98 ocupando completamente el disco duro disponible.

Con la idea de poder contar con los Sistema Operativos Windows y Linux en el mismo equipo, se inició la elección de cuáles de las distribuciones de Linux existentes podría ser la mejor. Realmente no fue fácil realizar la elección y se decidió por aquella versión que se pudiera conseguir fácilmente en algún centro de cómputo dedicado a la docencia y que permitiera realizar el desarrollo del presente trabajo.

Después de buscar en diferentes partes y de probar con diferentes distribuciones de Linux, se pudo finalmente conseguir el sistema operativo Windows XP y la distribución de Linux Red Hat 9 para equipos Intel. Ya con los dos sistemas operativos en la mano, se definió realizar la instalación de ambos sistemas operativos en el mismo equipo, para lo cuál se realizó la partición del disco duro de la siguiente forma: la primera partición quedó de casi 23 Gb y la segunda partición de 20 Gb; para realizar éstas particiones del disco duro hicimos uso de la paquetería para Windows *partition magic*.

En la partición dedicada a Linux, tuvimos que realizar una partición más para alojar el área de swap (área de intercambio de información), misma que debe tener un tamaño igual a dos o tres veces la cantidad de memoria RAM del equipo a utilizar. Pensando que se podría fácilmente ampliar la memoria RAM del equipo de 128 Mb a 256 Mb, ya que el equipo utiliza SIMS a 100 mhz (conocidos como pc100), generamos el área de swap de 500 Mb.

Las particiones realizadas al disco duro de 40 Gb, fueron las siguientes:



Revisaremos más adelante las características específicas de cada una de las particiones realizadas.

Pensando que más adelante, como se explicará, Linux tomaría control del arranque del sistema, se dejó en la primera partición al sistema operativo Windows Xp; de la mitad del disco hacia el final se tendrá instalado lo que corresponde al sistema operativo Linux y se dejó el área de intercambio (swap) al final de la segunda partición, con el firme propósito de evitar mover los sistemas operativos ya instalados.

Todo ello debido a que primeramente se formateo completamente el disco de 40 Gb y se reviso que no tuviera errores. En el caso de que se encontrara algún error se activo la opción de reparación automática.

Ya con el disco limpio y sin errores, se instaló el sistema operativo Windows Xp, con la idea que ocupará los primeros sectores del disco recién formateado. Se instaló también Microsoft office 2003, con el cuál se hizo la documentación del presente trabajo, y se crearon los usuarios del sistema.

Acto seguido, realizamos la instalación de Linux redhat 9 desde la lectora de CD, misma que detallaremos más adelante; el primer disco de Linux reconoció las particiones existentes, pudiendo entonces elegir la segunda partición del disco para la instalación, evitando así mover el sistema operativo Windows Xp que fue previamente instalado.

El área de swap se creó al final de la segunda partición del disco, pensando también en usar espacio del disco que hasta ese momento no había sido utilizado de forma alguna.

La intención es que Linux tome el control de arranque del equipo al encenderlo y se pueda elegir de cuales de los sistemas operativos instalados es el que deseamos utilizar. El arranque lo podemos realizar de dos diferentes formas, tal y como lo comentaremos más adelante en la sección de instalación y configuración.

Todo este esfuerzo fue realizado con la firme idea de poder tener en un mismo equipo dos muy diferentes sistemas operativos que puedan hacer uso de los diversos dispositivos que forman la Compaq Presario 5000.

Linux redhat 9 nos permitió realizar toda la programación del presente trabajo, mientras que Windows Xp facilitó en gran medida la realización de toda la documentación necesaria que permitiera sustentar tanto la hipótesis planteada, como la programación realizada.

Descarga del software

Linux redhat 9 esta compuesto por tres discos compactos, mismos que pueden ser verificados antes de inicial la instalación, hecho recomendable para excluir de esta forma posibles errores posteriores.

Inicializamos el equipo con el primer disco de Linux en la lectora de CD, es muy recomendable ya haber realizado las particiones necesarias del disco duro en el caso de que se quiera contar con otro u otros sistemas operativos en el mismo equipo; en caso de no haberlo hecho, podremos crear dichas particiones en el proceso de instalación.

El disco de arranque muestra una pantalla de bienvenida a red hat 9 y nos da las siguientes opciones a elegir:

- ✧ Instalar o actualizar redhat Linux en modo gráfico
- ✧ Instalar o actualizar redhat Linux en modo texto

Que es solamente la definición de cómo queremos que se realice la instalación, de las cuales seleccionamos la primera opción, para aprovechar el amigable ambiente que ofrece Linux durante su instalación.

Selección de Lenguaje

La primera pantalla que nos aparece ya en el proceso de instalación, es: *Language Selection* que nos permite definir el lenguaje que será utilizado durante la instalación.

Al lado derecho de la pantalla aparecerá una lista muy completa de idiomas que pueden ser seleccionados, tal y como se muestra en la imagen anterior.

Es recomendable seleccionar el idioma inglés, ya que muchos de los tecnicismos a utilizar pueden causar confusión si se presentan en español. Recordemos que nuestra área esta llena de palabras y conceptos creados en el idioma inglés y su traducción no es siempre muy fidedigna. Una vez realizada la selección, oprimimos el botón *next* (siguiente).

Todas las pantallas de instalación que utilizaremos y mencionaremos, tienen cuatro botones en la parte inferior de la misma, los cuáles facilitan el proceso de instalación, y son:

- *Hide Help*
  Con el que se esconde la ayuda durante la instalación, la cuál se encuentra en la parte izquierda de la pantalla.

- *Release notes*
  Permite mostrar las notas más recientes de la distribución utilizada.

- *Back*
  Regresa a la pantalla anterior.

- *Next*
  Avanza a la siguiente pantalla.

Selección del Teclado

La siguiente pantalla es: *keyboard configuration,* la cuál nos permitirá definir que tipo de teclado será utilizado en el sistema.

De la lista mostrada a la derecha de la pantalla, seleccionamos la opción: *spanish* (español). Una vez realizada la selección, oprimimos el botón *next*.

Selección del Ratón

A continuación aparecerá la pantalla: *Mouse Configuration*, donde definimos el tipo de ratón que se tiene en el equipo, incluso podemos emular un Mouse con tres botones.

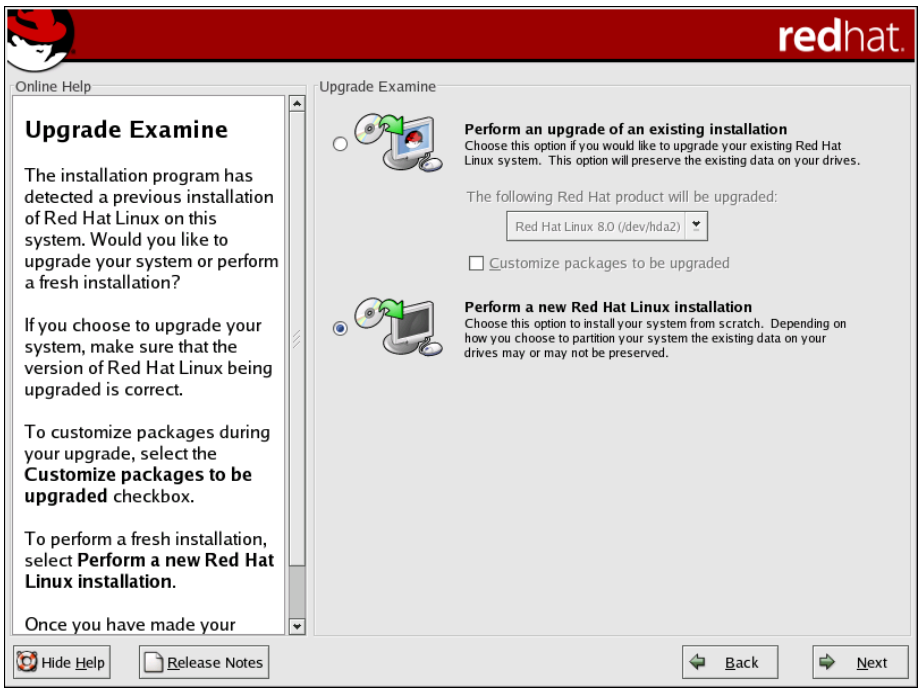
De la lista mostrada a la derecha de la pantalla, seleccionamos el *Wheel Mouse (PS/2)*, que es la sugerencia que da Linux de acuerdo al hardware que ha reconocido en el equipo. De hecho cuando se tenga duda de la selección a realizar, es conveniente tomar la sugerencia que da Linux durante la instalación.

Actualización o Instalación

La siguiente pantalla es: *Upgrade Examine,* donde debemos seleccionar una de las dos opciones mostradas, las cuáles son:

- ✧ Perform an upgrade of existing installation
- ✧ Perform a new Red Hat Linux installation

Tal y como se muestra en la siguiente imagen:



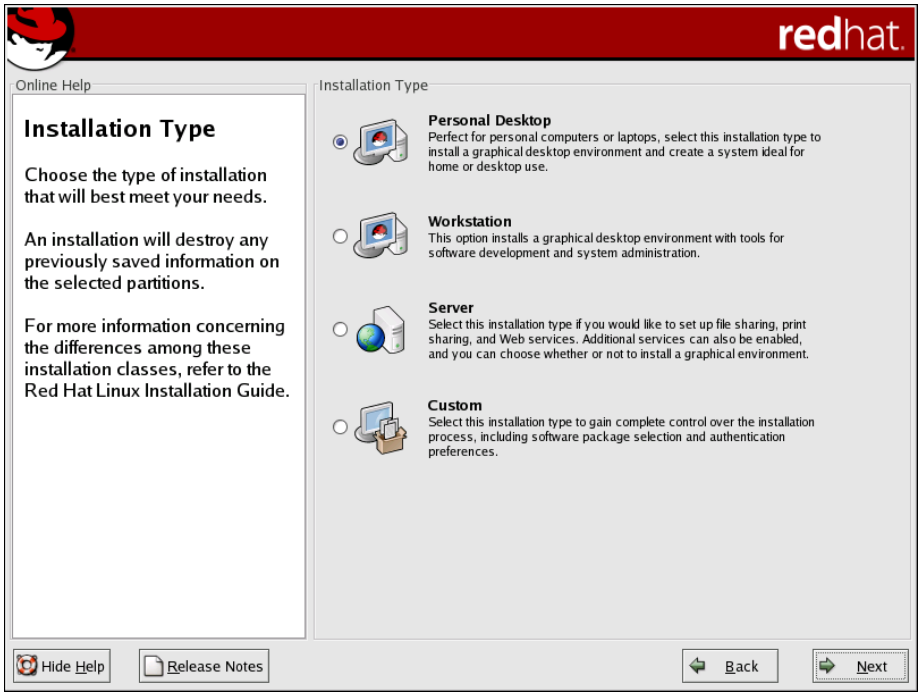
Como estamos realizando una nueva instalación, seleccionamos la segunda opción mostrada, la cuál es la más recomendable para que pueda formatear el espacio dedicado. Una vez realizada la selección, oprimimos el botón *next*.

Tipo de Instalación

A continuación aparecerá la pantalla: *Installation Type*, donde aparecen las siguientes opciones para definir el tipo de instalación que deseamos realizar:

- ✧ *Personal Desktop*
  Equipo personal de desarrollo.
- ✧ *Workstation*
  Estación de trabajo.
- ✧ *Server*
  Servidor de red o Internet.
- ✧ *Custom*
  Instalación personalizada.

La imagen que presenta Linux redhat es:



La opción dependerá de que uso se quiere dar a Linux, aunque lo recomendable es realizar una instalación personalizada (*custom*), con ello habrá que seleccionar todos y cada uno de los paquetes a ser instalados, incrementando el tiempo de instalación.

Partición del Disco

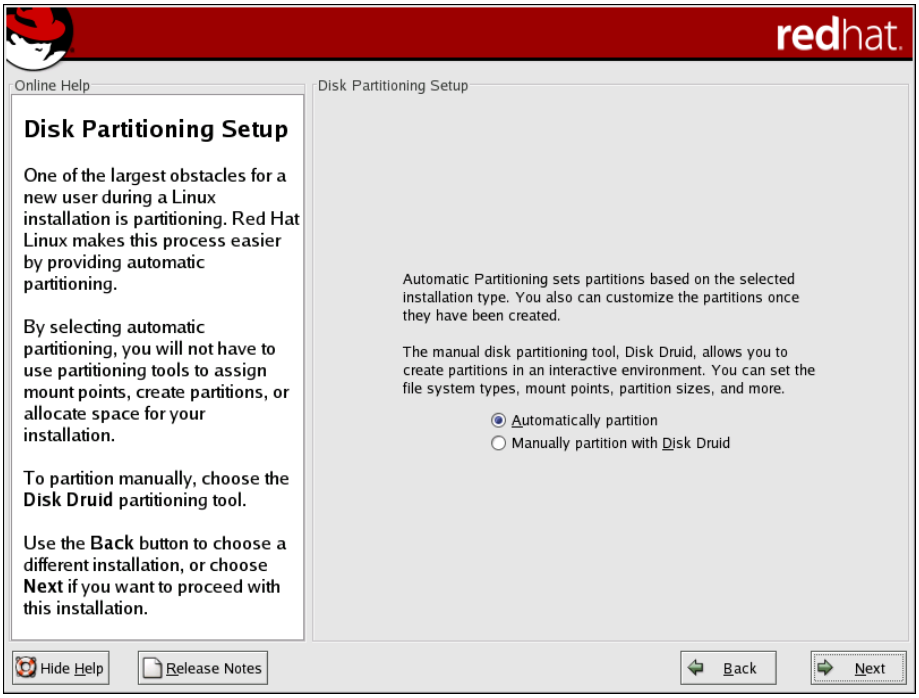
La siguiente pantalla en la instalación es: *Disk Partitioning Setup*, la cuál reconoce las particiones ya realizadas en el disco duro o permite crearlas en este momento.

Aunque como lo comentamos al inicio de ésta sección, no es lo más recomendable, ya que Linux querrá hacerlo por nosotros y probablemente no respete el sistema operativo Windows Xp que ya debió haber sido instalado en la primera partición del disco y que por ende debe dejarse tal y como se encuentra.

Las opciones que podemos elegir, son:

- *Automatically Partition*
- *Manually Partition*

La imagen que aparece y que nos permite definir la partición del disco, es:



Debemos seleccionar *Manually Partition*, para que se puedan reconocer las particiones ya realizadas antes de iniciar la presente instalación. En el supuesto caso de no tener realizadas las particiones y necesitar de dos o más sistemas operativos en el mismo equipo, se deberá seleccionar la opción: *Automatically Partition,* permitiendo que Linux las realice por nosotros. Una vez realizada la selección, oprimimos el botón *next*.

Definición del Disco

Una vez que se han detectado particiones en el disco, nos aparecerá la pantalla: *Disk Setup,* en donde se deben definir ciertas propiedades a cada una de las particiones encontradas, como es el tipo de archivo con los que son reconocidos por Linux.

Recordemos que en nuestro caso tenemos creadas las siguientes particiones en el disco duro:

/dev/

hda1 22,923 mb	hda2 19,505 mb	hda3 502 mb
htfs	ext3	swap

Device	Mount Point	Type	Format	Size (mb)
/dev/had				
/dev/had1	/boot	htfs		22923
/dev/had2	/	ext3	si	19505
/dev/had3		swap	si	502

En ésta pantalla podemos observar que el disco duro de 40 Gb que tiene instalado el equipo Compaq Presario 5000, es reconocido por Linux como: /dev/had, el cuál tiene tres diferentes particiones reconocidas como:

/dev/had1, Que es en donde se encuentra instalado el sistema operativo Windows Xp. Linux detecta que tiene el sistema de archivos htfs, con un tamaño de 22923 mb.

Aquí debemos definir el punto de arranque bajo la etiqueta *Mount Point*, el cuál deberá ser /boot, indicándonos así que el sector de arranque será alojado en la primera partición del disco y el cuál mostrará un *dual boot* (arranque dual), mismo que nos permitirá seleccionar con cuál de los dos sistemas operativos existentes se deberá inicializar el equipo.

Es MUY IMPORTANTE no formatear esta área del disco, ya que perderemos irremediablemente las aplicaciones que se hayan configurado a los usuarios de Windows Xp.

/dev/had2, Esta es el área destinada a Linux, por lo que el sistema de archivos seleccionado es ext3 y tendrá un tamaño de 19505 mb. Este sistema de archivos es el utilizado por las últimas versiones de redhat y es compatible con su predecesor ext2.

El punto de arranque será el directorio raíz del sistema operativo Linux, el cuál es root (/) y de donde se generará toda la estructura de directorios requerida por redhat 9.

Es importante formatear esta área para asegurar que la instalación de Linux no encontrara información alguna, por lo que se debe activar la opción en la pantalla bajo la etiqueta *Format*.

/dev/had3, Esta es el área destinada al intercambio de información en Linux, conocida como el área de swap y que se aconseja que tenga un tamaño igual a dos o tres veces el tamaño de la memoria RAM existente en el equipo.

Nótese que el tipo de archivo debe ser precisamente swap.

Linux redhat permite crear diferentes tipos de particiones, basadas en el sistema de archivos que cada uno utilice, manteniéndolas separadas e independientes una de las otras.

Los tipos de sistemas de archivos que se reconocen durante la instalación de Linux redhat, se definen a continuación:

- ext2
  El sistema de archivos ext2 soporta tipos de archivos estándar de Unix, como son archivos regulares, directorios, ligas simbólicas, etc. Permite manejar nombres de archivos de hasta 255 caracteres.
  Hasta las versiones de redhat 7.2 utilizan el sistema de archivos ext2 por omisión.

- ext3
  El sistema de archivos ext3 está basado en el ext2 y tiene una gran ventaja, llamada *journaling*, que le permite recudir el tiempo invertido en recuperar un sistema después de una catástrofe y en el cuál no es necesario realizar una revisión de la estructura de archivos.
  El sistema de archivos ext3 es recomendado en la instalación de Linux.

- swap
  Las particiones swap son requeridas para el uso de memoria virtual en el equipo. En otras palabras, los datos son escritos a particiones definidas en disco duro, cuando no hay suficiente espacio libre en la memoria RAM para almacenar los datos de algún proceso que se está ejecutando en ese momento.

- htfs
  Linux redhat 9 podrá reconocer cualquier sistema operativo Windows mediante el sistema de archivos htfs. Otras distribuciones de Linux no pudieron reconocer esta área tan fácilmente, produciendo que toda la instalación fracasara.

En las versiones más recientes de redhat sin duda existirán nuevos tipos de archivos soportados, ya que como sabemos, el soporte técnico está dado por los usuarios a nivel mundial mediante el uso de Internet, mismos que reportan y arreglan las necesidades identificadas.

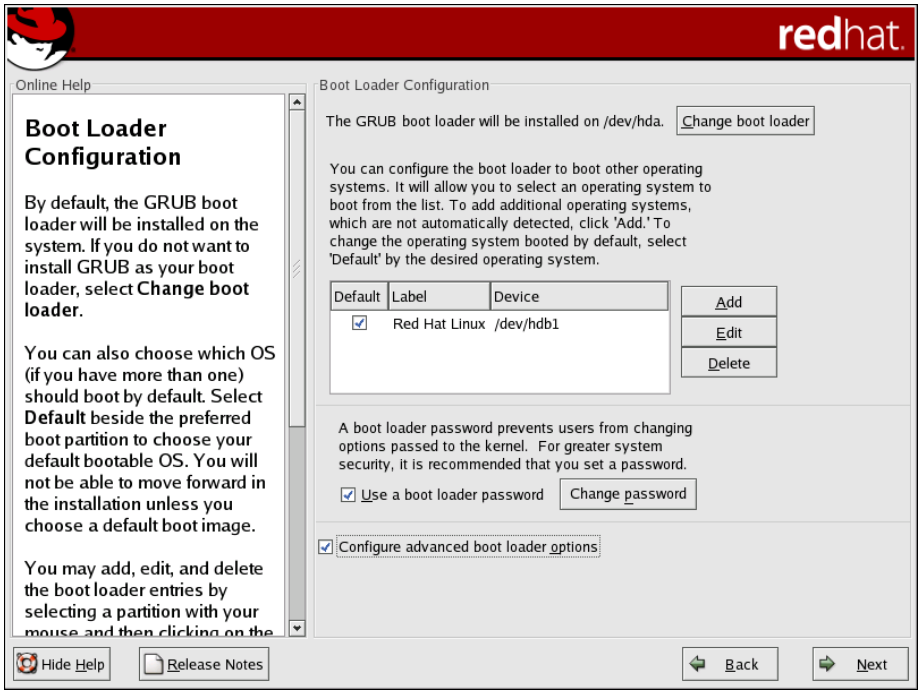
Como se comentó al inicio de ésta sección, es menester que ya se halla realizado la instalación del sistema operativo Windows Xp en una partición bien identificada del disco duro, para que sea reconocida pero nunca formateada, ya que el formateo ocasionaría la irreparable perdida de las aplicaciones configuradas en ese ambiente.

Sugerimos que se dejen los sistemas de archivos detectados y recomendados por Linux, ya que de alguna forma son los que podría y debería manejar.

Configuración de Arranque

La siguiente pantalla que nos aparece es: *Boot Loader Configuration*, en la cuál definiremos la configuración de arranque de nuestro sistema, por lo que debemos atender adecuadamente las indicaciones a seguir.

La imagen que aparece en la pantalla es:



Una vez que han sido reconocidas las particiones del disco, podemos seleccionar de cuál de ellas queremos inicializar el sistema, ya que seguramente tendrán sistemas operativos diferentes, como lo es nuestro caso.

Cada registro mostrado tendrá la etiqueta con la que es reconocida cada partición, así como el dispositivo en el que se encuentra instalado. En nuestro caso los dispositivos de arranque del sistema son: /dev/hda1 (Windows Xp) y /dev/hda2 (Linux).

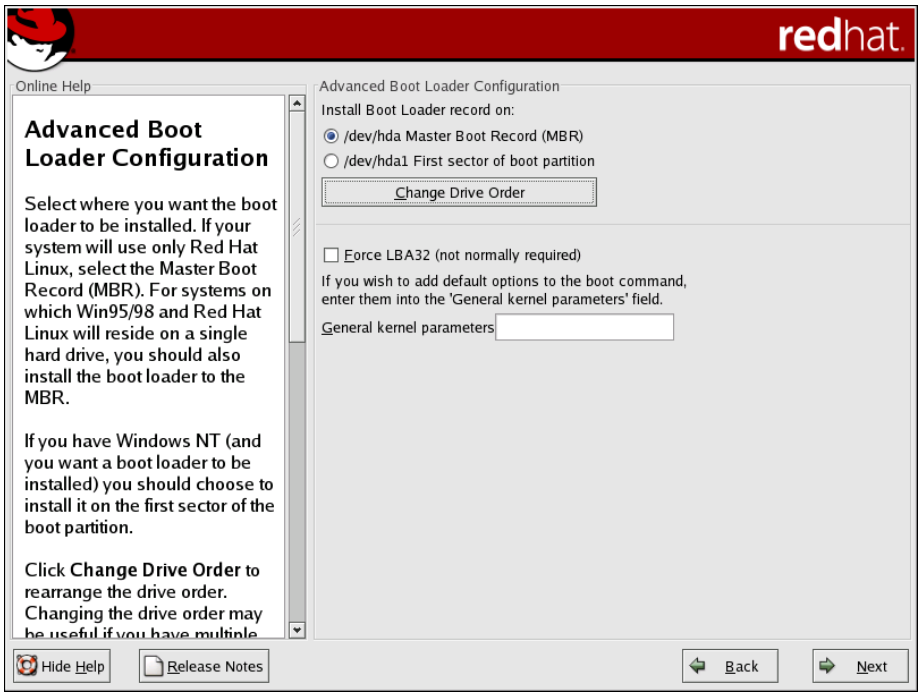
En la pantalla podremos agregar, editar (modificar) o borrar cualesquiera de los dispositivos mostrados. También podemos asignar una contraseña de inicio activando el *check box*: *use boot loader password*.

Para configurar opciones especiales o específicas de arranque del sistema, debemos activar el *check box*: *Configure advanced boot loader options*.
Oprimimos el botón *next*.

Configuración de Arranque Específica

Una vez seleccionada la opción: *Configure advanced boot loader options*, aparecerá la pantalla: *advanced boot loader configurartion*, que nos permitirá definir la forma en que se realizará el arranque del equipo.

Este punto debe realizarse con cuidado, ya que de ello depende que logremos el objetivo de poder inicializar el sistema con dos diferentes sistemas operativos. La imagen presentada por Linux redhat en la pantalla, se muestra en la siguiente página.



En esta pantalla podemos agregar otros dispositivos de arranque para el sistema, los cuáles ya debieron ser reconocidos.

Cada uno de ellos tendrá un nombre diferente, y por ende, rutas diferentes de arranque, tal y como lo definimos en la configuración de arranque.

Un *boot loader* es el primer archivo que se ejecuta cuando un equipo se arranca o enciende. Este programa es el responsable de ceder el control del equipo, y todos sus recursos, a uno u otro de los sistemas operativos existentes en las particiones del disco duro.

Se pueden definir dos diferentes tipos, los cuales son:

- ✧ GRUB (*GRand Unified Bootloader*)
- ✧ LILO (*Linux LOder*).

Cualquiera de los dos serán primordialmente alojados en el MBR (*Master Boot Redord*) del disco, aunque también podrían ser almacenados en el primer sector de la partición de arranque. Evaluando la forma en que se presentan los sistemas operativos encontrados, seleccionamos la opción: GRUB ya que es más versátil y tiene mejor presentación.

Para modificar el lugar en donde será grabado el *boot loader*, hacemos uso de ésta pantalla, en donde por omisión la opción siempre será MBR.
Debemos tener mucho cuidado en la selección a realizar, ya que no definir adecuadamente la forma y modo en que se realizará la carga del sistema, podría suceder

que el equipo no arrancara. En tal caótico caso, no olvidar que contamos con el disco de instalación, mismo que nos permitirá arrancar el sistema y realizar las adecuaciones necesarias para regularizar el problema.

No olvidar que como estamos inicializando el sistema operativo de la segunda partición del disco, la primera partición ya tiene instalado Windows Xp, por lo que debemos tener cuidado en no eliminar lo ya creado. Una vez realizadas las definiciones de arranque, oprimimos el botón *next*.

Configuración de Red

Para poder configurar las opciones de red, nos aparece la pantalla: *Network Configuration*, permitiéndonos la configuración del enlace de red utilizada por el equipo, tan indispensable en nuestros días.

Linux reconocerá los puertos existentes en el equipo y los denominará por: eth0, eth1, etcétera, en el caso de que se encuentre más de uno.

Para cada uno de éstos dispositivos detectados, debemos definir si se activaran en el momento de arranque del sistema, así como su respectiva dirección IP.
Una vez realizadas las definiciones de red, oprimimos el botón *next*.

Configuración de FireWall

La siguiente pantalla es: *Firewall Configuration,* permitiéndonos configurar el elemento situado entre la computadora y la red, utilizado para auditar toda la información que ingresa a nuestra computadora y al cuál se le ha denominado *fire wall* o pared de fuego.

Primeramente debemos definir el nivel de seguridad a utilizar, los cuáles pueden ser:

- ✧ Alto
- ✧ Medio
- ✧ Sin FireWall

Después debemos seleccionar si utilizamos las reglas por default de firewall (lo más recomendable) o las personalizamos de acuerdo a nuestras específicas necesidades. Una vez realizadas las definiciones del FireWall, oprimimos el botón *next*.

Selección del Lenguaje del Sistema.

Aparecerá la pantalla: *Additional Language Support*, en la cuál debemos seleccionar el lenguaje a utilizar en el sistema.

Linux redhat propone utilizar el idioma inglés como el lenguaje del sistema, aunque puede ser cambiado por cualesquiera de las múltiples opciones mostradas en la parte derecha de la pantalla. Aceptamos la propuesta de Linux y oprimimos el botón *next*.

Configuración de Zona

En la pantalla: Time *zone selection,* debemos definir la zona y, por ende, el uso horario en la que se encuentra nuestro equipo.

Seleccionamos la opción: América/México y presionamos el botón *next*.

Asignación de contraseña al administrador

En seguida, la pantalla: *set root password*, nos forza a asignar una contraseña a la clave de administrador de nuestro sistema.

Es de vital importancia establecer una contraseña a la clave de administrador del sistema, ya que podrá realizar cambios a voluntad y seguramente serán irreparables. La clave de administrador es el usuario root, y pertenece al grupo de administración.

Tecleamos la contraseña elegida en el campo *root password* y su confirmación en el campo siguiente, etiquetado como *confirm*. Si ambos campos coinciden, la contraseña para la clave del administrador será asignada. No está de más recordar que las contraseñas de las claves privilegiadas del sistema, se deben tener siempre a la mano y al mismo tiempo deben ser confidenciales.
Una vez asignada la contraseña del administrador oprimimos el botón *next*.

Autentificación de la Configuración

En seguida aparecerá la pantalla: *authentificarion Configuration*, para poder definir el tipo de contraseña que se utilizará.

Se pueden habilitar dos tipos de contraseñas:

- *MDS Password ,* el cuál permitirá el uso de contraseñas largas, es decir, de hasta 256 caracteres de longitud.
- *Shadow Password,* el cuál provee un método de seguridad para el acceso de los archivos de configuración del sistema

Seleccionamos la primera opción, incluso podemos seleccionar ambas, y oprimimos el botón *next*.

Selección de grupos de paquetes

En la pantalla: *Package Group Selection*, debemos seleccionar los paquetes o conjunto de aplicaciones a instalar en el equipo. Podemos seleccionar programas de desarrollo, editores, herramientas de administración, manejadores de bases de datos, aplicaciones

tipo Windows, programas de comunicaciones, herramientas para generar un servidor de Internet, etcétera.

Podemos verificar el contenido de cada uno de los paquetes de programas seleccionados, ya sea para agregar nuevos o eliminar elementos no deseados al paquete.
Cuando se selecciona el paquete podemos apreciar el número de programas que lo forman, incluso podemos revisar el detalle para realizar una selección aún más específica.

Si alguien lo prefiere, podría revisar todos los programas por lista, seleccionando la opción *Select Individual Packages* al final de la pantalla, presentándose así una gran lista de aplicaciones que pueden ser instaladas.

Consideramos que la selección por grupo es mucho más recomendable que la selección individual, ya que el grupo presenta herramientas afines enfocadas en una tarea específica. Si realizáramos la selección en forma individual, muy probablemente faltaría algún elemento importante. De hecho antes de realizar la instalación de los paquetes, Linux realiza una revisión de dependencias entre los programas que han sido seleccionados para la instalación, pantalla que describiremos a continuación.

Una vez que estamos conformes con la selección realizada, oprimimos el botón: *next*.

Revisión de Dependencias

Una vez realizada la selección de los paquetes a instalar, Linux revisa las dependencias entre ellos y presenta en la pantalla: *Unresolved Dependencies,* para aquellas aplicaciones que requieren de otro elemento a instalar y que no ha sido seleccionado para la instalación.

Ante esta situación podemos hacer una de tres cosas:

- ✧ Realizar la instalación de los paquetes faltantes para satisfacer las dependencias
- ✧ No instalar los paquetes que tienen dependencias incompletas
- ✧ Ignorar las dependencias entre los paquetes

Es siempre preferible seleccionar la primera opción en caso de que se presenten dependencias incompletas entre los paquetes que deseamos instalar.

Instalación de paquetes

Para iniciar propiamente con la instalación de las aplicaciones que han sido seleccionadas, aparecerá la pantalla: *installing Packages*.

Una vez completada la selección de todos y cada uno de los programas que formarán nuestro ambiente Linux, y de los cuales se hayan resuelto todas las dependencias entre sí, se inicia su instalación propiamente en nuestro equipo.

Dependiendo de la selección grupal o individual de programas (los cuáles están agrupados en paquetes), tendremos finalmente un determinado número de programas a instalar.

La imagen mostrada en pantalla por redhat, se ilustra a continuación.

Los paquetes están generados en formato RPM (*redhat Package Management*), que es un seguro formato de encripción de paquetes de información creado por redhat, que se ha ido convirtiendo en un estándar entre las diferentes distribuciones de Linux.

Podemos observar que la pantalla de instalación de paquetes nos informa el número total de programas a instalar, el tamaño de los mismos y el tiempo estimado de instalación, que será actualizado regularmente por el mismo instalador, por lo que el tiempo mostrado al inicio de la instalación seguramente será mayor al requerido.

Siempre podremos saber que programa se está instalando, su tamaño y su descripción. También podremos saber cuantos programas han sido instalados, cuantos faltan por ser instalados y, el tamaño y tiempo estimado de instalación de ambos. El tiempo requerido para una instalación completa, digamos 600 diferentes programas, llevó casi tres horas en la Compaq Presario 5000, que como sabemos, no tiene un procesador muy potente y por ende debimos ser pacientes.

Creación de disco de arranque

Una vez realizada la instalación de los paquetes seleccionados, aparecerá la pantalla: *Boot Diskette Creation,* la cuál nos sugiere crear un disco de arranque del sistema recién instalado.

Es altamente recomendable crearlo, ya que pude sernos de vital ayuda en casos de catástrofe.

Para ello debemos tener preparado un disco flexible recién formateado y bien etiquetado, ya que cuando se requiere no podemos encontrarlo entre la gran cantidad de discos flexibles con los que contamos.

Colocamos el disco flexible en el drive correspondiente y seleccionamos la opción: *Yes, i World like to create a boot diskette*. Es importante mencionar que este disco flexible de arranque del sistema puede ser creado después de completar la instalación de Linux, por si alguien olvido realizarlo.

Linux puede arrancar el sistema tanto en modo gráfico (que es el default) o en modo carácter. Para poder inicializarlo en modo carácter, debemos editar el archivo: /etc/inittab

En este archivo, que se encuentra muy bien documentado y que define la forma en que se realizará el encendido del equipo, debemos cambiar a 3 el número 5 dentro del rubro inicialización; nótese que el archivo tiene todos los rubros con los valores definidos durante la instalación que acabamos de realizar.

Este archivo será, sin duda, el primero que será revisado una vez que sea utilizado el disco flexible de arranque en casos de catástrofe.

Hay refranes que nos recomiendan el que es mejor tenerlo y tal vez nunca utilizarlo, que llegar a necesitarlo sin que lo tengamos, por lo que no debemos olvidar generarlo cada vez que creemos una nueva instalación o actualización de Linux.
Oprimimos el botón *next*.

Configuración de video

Para poder configurar adecuadamente el video, lo cual es esencial, aparecerá la pantalla: *Graphical Interface (X) Configuration.*

Como en todo sistema gráfico, el reconocer los dispositivos de salida es factor de suma importancia; Linux nos sugiere elegir aquel que ha detectado, aunque como siempre, podremos seleccionar aquel que convenga a nuestras necesidades, sin olvidar que el hacer una mala elección repercutirá en el no poder hacer uso adecuado del video de nuestro sistema y todo lo que ello conlleva.

En esta pantalla podemos adecuar la cantidad de RAM que será dedicada al uso del video, misma que está definida en la tarjeta a utilizar.

En el supuesto caso que querer recuperar los valores sugeridos por el instalador, bastará con oprimir el botón: *Restore original values*.
Una vez que estamos de acuerdo con las opciones elegidas, oprimimos el botón *next*.

Continuación de la Configuración de video (2 de 3)

Para continuar con la correcta configuración del monitor, aparecerá la pantalla: *Monitor configuration*.

Linux reconoce y optimiza el uso del monitor detectado, obteniendo la velocidad de barrido tanto horizontal como vertical, mismas que pueden ser modificadas haciendo uso de los campos: *horizontal y vertical sync*. Estos valores están definidos en la documentación del monitor e incluso en la parte posterior del mismo, pero no es necesario consultarlos, ya que Linux lo ha hecho por nosotros.

En caso de querer restablecer los valores detectados por el instalador, deberemos oprimir el botón: *Restore original values*.
En cuanto estemos de acuerdo con la selección realizada, oprimimos el botón: *next*.

Continuación de la Configuración de video (3 de 3)

Finalmente, para definir la resolución a utilizar en el video, aparecerá la pantalla: *Customize Graphics Configuration*.

Con la idea de poder maximizar el uso del video detectado por el instalador, Linux propone la resolución a utilizar, misma que puede ser modificada cambiando los valores de los campos: *color depth y screen resolution*.

Color depth es el número de colores diferentes que puedes ser utilizados por las aplicaciones y *screen resolution* es el número de *pixeles* que pueden ser utilizados en la pantalla.

Al final de la pantalla, encontramos la pregunta:

Please choose your login type: Graphical or text

Esta pregunta nos hace recordar el ya mencionado archivo: /etc/inittab, en el cuál encontraremos secciones con los valores de inicialización definidos para la sesión, mismos que pueden ser modificados utilizando alguno de los varios editores de texto incluidos con redhat 9. Claro está que siempre podrá ser utilizado el legendario vi (visual editor).

La inicialización del sistema en forma gráfica, que es el default, se realizará mediante el comando: init 5. Recordemos que este comando esta inmerso en el archivo /etc/inittab que es el primer archivo en ser ejecutado por el sistema al arrancar. Si quisiéramos realizar el arranque en modo carácter, que es el clásico de Unix, deberemos modificar el argumento del comando, usando: init 3 para modo multiusuario e init 1 para monousuario.

Para realizar el arranque en un nuevo modo, es necesario detener la ejecución del modo actual, mediante los comandos:

```
sync
shutdown now
```

Con los cuales se escribe a disco las últimas actualizaciones realizadas y detiene el uso del sistema. No olvidar que estos comandos solamente pueden ser ejecutados por el administrador del equipo. De ésta forma, la instalación ha terminado satisfactoriamente y estamos listos para hacer uso de Linux redhat 9.

Comprobación de arranque.

Al reiniciar el equipo, y si todo fue realizado correctamente sin contratiempos, el *boot loader* generado presentará en la pantalla los dos sistemas operativos con los que puede ser inicializado, como se muestra en la siguiente fotografía.

El *boot loader* presenta el orden definido para inicialización del equipo, dando diez segundos para realizar el cambio de opción; en caso de no tener respuesta elige la primera opción configurada. En nuestro caso la primera opción arranca el equipo con el sistema operativo Windows Xp y la segunda opción inicializa el equipo con Linux redhat 9, tal y como lo definimos desde un principio, ya que así realizamos las particiones del disco duro.

Seleccionemos Linux redhat 9 y podremos observar la validación que se realiza a cada uno de los dispositivos definidos en el sistema operativo. Cada vez que se realiza una verificación exitosa del dispositivo detectado, Linux lo marca con la etiqueta OK en color verde.

En caso de que alguno de los dispositivos detectados sea marcado con la etiqueta FAIL, en color rojo, debemos revisar cuidadosamente el dispositivo en cuestión una vez que ha arrancado el equipo, en caso de que éste arranque, ya que de no hacerlo, deberemos hacer uso del disco flexible generado durante la instalación y resolver así el problema detectado. En el peor de los casos, deberemos iniciar nuevamente la instalación del sistema operativo, sin olvidar formatear el espacio destinado a Linux y deberemos poner más atención en cada uno de los pasos de la instalación.

Enseñar es ante todo y sobre todo, aprender.
Miguel de Unamuno.

4. Distribuciones existentes en el mercado

Gracias a la apertura del código fuente del sistema operativo Linux, surgieron una gran variedad de matices, dando paso a las distribuciones que se pueden obtener de este singular sistema operativo.

Podemos decir que una distribución de Linux es un conjunto de aplicaciones y herramientas de instalación y configuración, las cuales operan en conjunto con el núcleo o *kernel* del sistema operativo.

Actualmente existe una amplia variedad de distribuciones de Linux, algunas están enfocadas a propósitos muy específicos (como son seguridad y servidores de alto rendimiento) y otras de propósitos generales (como por ejemplo, una computadora de escritorio).

El software que es parte de la distribución es elegido por sus creadores, los cuales podrían ser una compañía como redhat, una comunidad como Debian, o una universidad de algún país, etcétera.

La mayoría de las distribuciones ofrecen paquetes precompilados, como puede ser RPM (*RedHat Package Management*), haciendo sencillo la instalación de nuevo software en el sistema, incluso existen herramientas que nos facilitan el mantener al día nuestro sistema, tales como APT de Debian, up2date de redhat o Red Carpet de Ximian.

Hacer una lista de todas las distribuciones existentes de Linux no es una tarea fácil. Para hacerlo más sencillo, presentamos a continuación la lista de las distribuciones más populares en el mercado.

RedHat redhat es la distribución más conocida y usada en el mundo, la compañía fue fundada en 1994 y además de dedicarse a la producción de la distribución ofrece otros servicios como lo son la *RedHat Network* o las certificaciones como RHCE (*RedHat Certified Engineer*). Es por esto que redhat es ampliamente aceptada en la industria de la tecnología informática (TI). El presente trabajo está realizado utilizando la distribución Linux redhat 9. La dirección electrónica de su site es: http://www.redhat.com/	

Mandrake

MandrakeSoft fue creado en 1998 con el propósito de construir un Linux más fácil de usar para cualquiera.

Nació como una distribución basada en redhat, añadiendo algunas características que no estaban integradas, como el entorno gráfico KDE y un instalador gráfico simple y sencillo de usar.

Mandrake es ideal para usuarios nuevos que no desean involucrarse con profundos conocimientos técnicos, debido a su facilidad de uso.

La dirección electrónica de su site es:

http://www.mandrakesoft.com/

LGIS Linux

LGIS GNU/Linux 9 es una versión modificada de redhat 9 (Shrike). LGIS GNU/Linux 9 es una distribución orgullosamente mexicana, la cual surge de la necesidad de contar con la última versión de la distribución más utilizada a nivel mundial, con todas sus actualizaciones, además de la inclusión del Escritorio Ximian Desktop 2 (XD2) con todos los productos libres (Evolution, RedCarpet, etc.) lo que permite entre otras cosas mantener el sistema actualizado al 100% y manipular toda la información personal con la mejor herramienta para ello (Evolution), además, la versión Ximianizada de OppenOffice.org la Suite de Oficina libre que está reemplazando a sus contrapartes propietarias.

La dirección electrónica de su site es:

http://www.lgislinux.com/

LinuxPPP

Linux PPP (*Proyecto Personal de Pepe*) es junto con LGIS Linux de las únicas distribuciones mexicanas existentes en el mercado.

Fue una de las distribuciones de Linux más utilizadas en México y la única con influencia en toda Latinoamérica y España.

Actualmente se encuentra en estado de desarrollo puesto que su última versión fue LinuxPPP 6.4, que se encuentra basada en redhat 6.2.

La dirección electrónica de su site es:

http://linuxppp.com/

Debian

El proyecto Debian nació en 1993 como una organización de individuos que tienen como causa común crear un sistema operativo 100% libre.

Debian Linux es una distribución completamente libre alejada de todo tipo de asociación comercial y de software propietario alguno.

Su desarrollo por parte de programadores de todo el mundo es uno de los más grandes llevados a acabo por la comunidad de software libre.

Más allá del aspecto técnico Debian se acerca al fundamento de *free software*, que dio nacimiento a Linux.

La dirección electrónica de su site es:

http://www.debian.org/

Knoppix

Knoppix es una distribución de origen alemán, basada en Debian, que tiene una característica muy especial, la cual es que se debe ejecutar directamente del CD sin necesidad de instalarlo en el disco duro.

Por tal razón puede ser usado como una herramienta de recuperación o bien para echarle un vistazo a Linux antes de instalarlo.

La dirección electrónica de su site es:

http://www.knoppix.org/

Slackware

Slackware fue la primera distribución de Linux como las conocemos hoy en día. Su filosofía es mantener absolutamente todo sencillo (KISS) tomando muchas ideas de los UNIX originales, tales como el sistema de arranque.

Muchos usuarios prefieren Slackware precisamente por esa sencillez, la instalación es basada en texto y es tan sencilla que un columnista de la *Linux Journal Magazine* comentó que podría completar una instalación de Slackware sin un monitor conectado a la computadora. Parte de esta sencillez es la carencia de sistemas automatizados de configuración, sin embargo incluye un sencillo sistema de paquetes. Excelente como puente entre Linux y sistemas BSD tanto para usuarios avanzados como para novatos.

La dirección electrónica de su site es:

http://www.slackware.org/

Linux From Scratch

Linux From Scratch (LFS) es un proyecto que consiste en proveer los pasos necesarios para poder construir desde cero una distribución propia.

Es una de las mejores maneras de conocer cómo funciona un sistema Linux por dentro, así como conocer la relación entre los componentes del sistema.

La dirección electrónica de su site es:

http://www.linuxfromscratch.org/

Gentoo

Gentoo Linux es una distribución de reciente creación basada en código fuente, es decir, provee en conjunto con su sistema de paquetes, una jerarquía de instrucciones que automatiza la descarga, compilación, actualización y empaquetado de software en la computadora personal.

Esto permite optimizar, configurar y *mantener al día* la computadora, es decir, permite el personalizarla por completo.

Esta distribución es ideal tanto para novatos que deseen conocer su sistema Linux a fondo como administradores de red, programadores y usuarios de Slackware o Linux from Scratch.

La dirección electrónica de su site es:

http://www.gentoo.org/

SuSE

SuSE se enfoca al mercado de los escritorios, y es famoso por ser muy fácil de instalar y por su herramienta de configuración llamada YaST.

El desarrollo de SuSE es un tanto cerrado ya que no proveen versiones beta de su distribución y además no colocan imágenes ISO para descargar la distribución desde Internet, lo que es muy común en otras tantas distribuciones existentes.

La dirección electrónica de su site es:

http://www.suse.com/latam/

Lycoris Desktop/LX

Destkop/LX es un sistema operativo basado en Linux hecho por la compañía Lycoris la cual proclama ser el proveedor líder de Linux específicamente orientado al mercado del escritorio, lo cuál podremos constatar en poco tiempo.

Junto con *Lindows,* ha sido una de las distribuciones que han sido preinstaladas en computadoras vendidas en la cadena de tiendas WallMart, intentando con ello lograr una mejor penetración de mercado al clásico y recurrente estilo norteamericano.

La dirección electrónica de su site es:

http://www.lycoris.com/

Lindows

Lindows es una distribución dirigida al consumidor, con un *look and feel* al de Microsoft® Windows XP® o Apple® MacOS X®, lo cual incluye soporte para ejecutar aplicaciones tanto de Microsoft® Windows® como de Microsoft® Office®.

Esta distribución no es de libre acceso, pues tiene un costo por licencia y por ende cuenta con soporte técnico especializado por Internet.

La dirección electrónica de su site es:

http://www.lindows.com/

Xandros

Xandros es una distribución canadiense que está basada en Corel® Linux.

Esta enfocado en crear una solución de escritorio que permita combinar lo mejor de las tecnologías de código abierto con una atención corporativa hacia el soporte y usabilidad, así como la completa compatibilidad con software de Microsoft® Windows®, tan utilizado hoy en día.

La dirección electrónica de su site es:

http://www.xandros.com/

Como lo comentamos al inicio de esta sección, estas son las distribuciones más populares de Linux existentes en el mercado, por lo que hace falta mencionar algunas más que no son aun tan conocidas, pero que seguramente pronto lo podrán ser, debido al desarrollo constante en el que se encuentran.

Es importante señalar que toda la información relativa a esta sección fue consultada y puede ser cotejada, actualizada y ampliada en la página oficial electrónica de la comunidad GNU/Linux en México, que son por mucho, los que consideramos expertos en la materia en nuestro país.

Al ingresar a su página, encontramos el siguiente mensaje:

"GNU/Linux es hoy en día, la expresión más popular del software libre, una nueva dimensión, un concepto revolucionario en el desarrollo del cómputo. Esta nueva revolución tecnológica, brinda ahora al usuario y desarrollador de computadoras, total control y legalidad sobre el software que ejecuta, al substituir las tecnologías existentes, con reemplazos libres de barreras en costos, formas de uso, con formatos abiertos, seguridad y acceso total al funcionamiento interno de todos los sistemas que se usen."

La dirección electrónica del site y el singular logo de la comunidad GNU/Linux en México, a la cuál pertenecemos, son:

linux.org.mx

Nuestra elección

Después de revisar algunas de las distribuciones disponibles en el mercado, ya que llevaría demasiado tiempo el probar y evaluar todas y cada una de ellas, decidimos utilizar Linux redhat 9 por que esta muy bien posicionado en diferentes centros de cómputo dedicados a la docencia, así como en diferentes laboratorios de atención a alumnos, incluso, dentro de la misma Facultad de Ingeniería de la UNAM.

Además redhat es un producto diferente, ya que ofrece y reconoce dos diferentes certificaciones de conocimiento del mismo, las cuales son: RHCT (*RedHat Certified Technical*) y RHCE (*Red Hat Certified Engineer*), con los cuáles se puede obtener un reconocimiento ya sea como técnico o ingeniero, dependiendo de la profundidad que se tenga de él.

Somos lo que hacemos día a día, de modo que la excelencia no es un acto sino un hábito.
Aristóteles

5. Razones por las que Linux es tan utilizado hoy en día

Desde su creación, Unix mostró tener características únicas en comparación con otros sistemas operativos contemporáneos. Estas características son las que le dieron un lugar muy especial en el área de la informática y entre otras se encuentran el poder atender simultáneamente a varios usuarios (multiusuario), el poder portar el sistema operativo a casi cualquier plataforma (multiplataforma), la versatilidad de atender varios procesos concurrentes (multitareas), al mismo tiempo ser un sistema muy robusto ya que puede solucionar problemas no contemplados, por lo que es tan utilizado en aplicaciones de tiempo real.

Además cuenta con un lenguaje de comandos muy potente y de fácil programación, conocido como programación shell. Como su núcleo (*kernel)* está escrito en lenguaje C, tiene una conexión casi transparente con este potente lenguaje de programación, lo que facilita la generación de aplicaciones.

Desde que se volvió tan popular, Unix fue portado a diferentes equipos y cada uno de los fabricantes de hardware plasmo su sello en el nombre del sistema operativo, surgiendo de ésta forma lo que actualmente se conoce como los diferentes “sabores” de Unix, ya que se tradujo literalmente del adjetivo *flavors* en inglés.

Entre los principales “sabores” que se cuenta de Unix actualmente, se tienen los siguientes:

- En equipos Hewlett Packard, tomó el nombre de HP-UX.
- En equipos SUN, se le denominó Solaris.
- En equipos IBM, se le conoce como AIX.
- En equipos Macintosh, se le bautizó como AUX.
- En equipos Digital, se le nombró ULTRIX.
- En equipos Altos, se le conoce como XENIX.
- En computadoras personales, se creó Linux, de ahí la idea que es un Unix para computadoras personales.

Actualmente, Linux posee todas las características que pueden encontrarse en cualquier sistema Unix moderno, incluyendo direccionamiento lineal de 32 bits, memoria virtual,

multitarea real, *shared libraries*, módulos de *kernel* cargables on-demand, soporte TCP/IP (incluyendo SLIP, PPP, NFS, etc.) y entorno gráfico X-Windows.

Linux es distribuido bajo la Licencia General Pública de GNU, lo cual significa que puede ser distribuido, copiado y modificado gratuitamente, a condición de no imponer ninguna restricción en sucesivas distribuciones. En pocas palabras, Linux es un sistema operativo gratuito.

Componentes de Linux

El sistema operativo Linux se divide generalmente en cuatro componentes principales, los cuáles son:

- ✧ el núcleo (*kernel*)
- ✧ el shell
- ✧ el sistema de archivos
- ✧ las utilidades

El núcleo o kernel

Es el programa medular que ejecuta programas y gestiona dispositivos de hardware tales como los discos y las impresoras.

El shell

Proporciona una interfaz para el usuario y recibe órdenes del usuario (comandos) y las envía al núcleo para ser ejecutadas.

El sistema de archivos

Organiza la forma en que se almacenan los archivos en dispositivos de almacenamiento tales como los discos. Los archivos están organizados en directorios. Cada directorio puede contener un número cualquiera de subdirectorios, cada uno de los cuales puede a su vez, contener otros archivos.

El núcleo, el shell y el sistema de archivos forman en conjunto la estructura básica del sistema operativo. Con estos tres elementos primordiales, Linux puede ejecutar programas, gestionar archivos e interactuar con el sistema. Además, cuenta con unos programas de software llamados utilidades que han pasado a ser considerados como características estándar del sistema.

Las utilidades

Son programas especializados, tales como editores, compiladores y programas de comunicaciones, que realizan operaciones de computación estándar. Incluso cada uno de nosotros podemos crear nuestras propias utilidades.

Linux contiene un gran número de utilidades. Algunas efectúan operaciones sencillas, otras son programas complejos con sus propios juegos de órdenes. Para empezar, muchas utilidades de pueden clasificar en tres amplias categorías:

- Editores
- filtros y
- programas de comunicaciones.

También hay utilidades que efectúan operaciones con archivos y administración de programas.

En líneas generales podemos decir que se dispone de varios tipos de sistema de archivos para poder acceder a archivos en otras plataformas. Incluye un entorno gráfico X window (Interface gráfico estandard para máquinas Unix), que nada tiene que envidiar a los modernos y caros entornos comerciales. Está orientado al trabajo en red, con todo tipo de facilidades como correo electrónico por ejemplo.

Posee cada vez más software de libre distribución, que desarrollan miles de personas a lo largo y ancho del planeta azul. Linux es ya el sistema operativo preferido por la mayoría de los informáticos.

Un ejemplo de la popularidad que ha alcanzado es sistema y la confianza que se puede depositar en él, es que incluso la NASA ha encomendado misiones espaciales de control de experimentos a la seguridad y la eficacia de Linux.

La gran popularidad que se ha ganado Linux entre sus múltiples usuarios, se basa principalmente en los siguientes aspectos:

- Se distribuye su código fuente, lo cual permite a cualquier persona que así lo desee hacer todos los cambios necesarios para resolver problemas que se puedan presentar, así como también agregar funcionalidad. El único requisito que esto conlleva es poner los cambios realizados a disposición del público.

- Es desarrollado en forma abierta por cientos de usuarios distribuidos por todo el mundo, los cuales hacen uso de Internet como medio de comunicación y colaboración. Esto permite un rápido y eficiente ciclo de desarrollo.

- Cuenta con un amplio y robusto soporte para comunicaciones y redes, lo cual hace que sea una opción atractiva tanto para empresas como para usuarios individuales.

- Da soporte a una amplia variedad de hardware y se puede correr en una multitud de plataformas: PC's convencionales, computadoras Macintosh y Amiga, así como costosas estaciones de trabajo de diferentes fabricantes de hardware.

Linux es multitarea

La multitarea no consiste en hacer que el procesador realice más de un trabajo al mismo tiempo (un solo procesador no tiene esa capacidad), lo único que realiza es presentar las tareas de forma intercalada para que se ejecuten varias simultáneamente.

Por lo tanto en Linux es posible ejecutar varios programas a la vez sin necesidad de tener que parar la ejecución de cada aplicación.

Linux es multiusuario

Para que pueda desarrollar esta labor (de compartir los recursos de una computadora) es necesario un sistema operativo que permita a varios usuarios acceder al mismo tiempo a través de terminales, y que distribuya los recursos disponibles entre todos.

Así mismo, el sistema debería proporcionar la posibilidad de que más de un usuario pudiera trabajar con la misma versión de un mismo programa al mismo tiempo, y actualizar inmediatamente cualquier cambio que se produjese en la base de datos, quedando reflejado para todos.

Linux es multiplataforma

Es decir, que puede ser ejecutado en muy diferentes equipos como son Intel, AMD, motorola, sun, sparc, etcétera.

Linux es seguro

Linux se autoprotege, ya que no existen virus para este sistema operativo.

Recordemos que en la estructura de Linux aparecen cuatro elementos situados en bloques diferentes, cada uno de los cuales tiene encomendados una función y éstos son:

- Usuario
- Shell
- Núcleo o Kernel
- Hardware

El usuario utiliza todas las aplicaciones desarrolladas para Linux, así como sus utilidades y ordenes propias. Su información estará organizada mediante un sistema de archivos.

De todo el sistema solo puede interactuar con el shell (que es el encargado de interactuar entre el usuario y el kernel) que a su vez es el interprete de comandos, así como un lenguaje robusto de programación con el que es posible programar nuevas funciones o personalizar algunas existentes.

El núcleo es el único que interactúa con el hardware. Incluye entre sus funciones las operaciones más importantes de gestión del sistema operativo, como puede ser la gestión de memoria, mantenimiento del sistema de archivos, asignación de tiempo del CPU a cada una de las tareas, el control del acceso a un ordenador mediante claves, etcétera.

El elemento hardware estará formado por todos los componentes que en un momento determinado puede detectar el sistema.

Por todas estas cualidades tan especiales que reúne el sistema operativo, fue y sigue siendo muy utilizado en sistema de tiempo real, en las cuales el tiempo de respuesta es crítico.

Algunas de las áreas en las que actualmente Linux es muy utilizado, y seguramente lo seguirá siendo, son:

- ✓ ATM´s (*automatic teller machines, cajeros automáticos*)
  La gran mayoría de los cajeros automáticos existentes, cuentan con un servidor Unix que realiza las transacciones directamente en el cajero automático y actualiza bases de datos propietarias residentes en Main Frames, vía batch al final de la jornada de trabajo.

- ✓ Servidores de Internet
  Los principales servidores de Internet en la red de redes tienen instalado algún *sabor* de Unix, haciéndo de esta forma más versátil y confiable su uso y no su abuso.

- ✓ Bases de datos locales y distribuidas
  Por las bondades que ofrece Linux, se ha convertido en nicho de los principales manejadores de bases de datos relacionales, permitiendo crear tanto bases de datos locales como distribuidas.

- ✓ Arquitectura Cliente / Servidor
  Unix y la aparición de las computadoras personales vinieron a desplazar la idea generalizada de tener terminales *tontas* conectadas a un solo computador central que realizara todas las tareas encomendadas, compartiendo así los recursos existentes.

De todas las anteriores y de las cuáles se podría hablar mucho, nos gustaría profundizar un poco en la muy conocida y mencionada arquitectura cliente/servidor, que como explicaremos, permitió generar una nueva forma de procesamiento de información y de la cuál existen dos modalidades primordiales en el mercado.

Arquitectura Cliente / Servidor

Este sistema operativo permitió iniciar un concepto revolucionario, el cuál tenía como principio utilizar los recursos propios de las tan populares e incipientes computadoras personales y centralizar las bases de datos utilizadas en un solo servidor (de ahí su nombre) que atendiera las diferentes peticiones de todas esas computadoras personales con las que puede dialogar.

La nueva arquitectura, de ese entonces, vino a desplazar la tradicional idea de contar con un determinado número de terminales (conocidas después como *tontas*) conectadas a uno y sólo un computador central que se encargaba de realizar todas las tareas y de atender las peticiones de las diferentes terminales que reconocía.

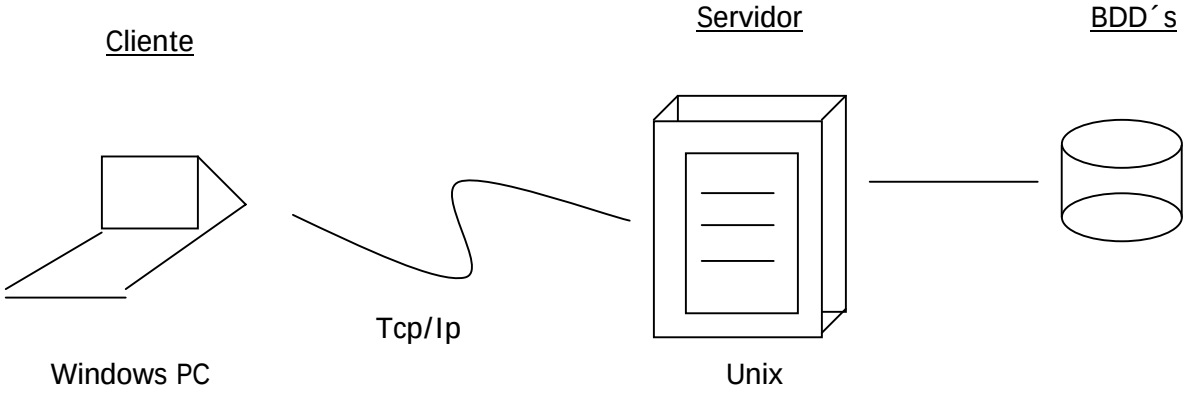
La arquitectura fue bautizada con el nombre de cliente-servidor, precisamente por la idea primordial, misma que estaba sustentada en poder separar las operaciones que debían ser realizadas en donde se producen las consultas y en donde se mantiene la información.

Todo esto sin duda fue posible con el gran auge que tuvieron y que siguen teniendo las computadoras personales, mejor conocidas como PC´s.

El creciente y demandante mercado de la informática, el cuál intenta tener y mantener la información vigente y disponible para todos sus usuarios, sigue mejorado los procesos que se han creado con tal fin; de ésta forma la arquitectura cliente-servidor conceptualizada inicialmente de dos capas, se ha ido modificando para adaptar una nueva forma bautizada como de tres capas.

a) De dos capas

La arquitectura cliente/servidor de dos capas, está conceptualizada en el siguiente diagrama:



En el diagrama mostrado podemos claramente apreciar tres elementos indispensables, los cuáles son:

- El Cliente
- El Servidor
- El medio de comunicación (la Red)

Las características primordiales de cada uno de estos elementos las describiremos a continuación.

El Cliente

Con la clara idea de poder aprovechar los grandes recursos existentes en las computadoras personales y con la creación de sistemas operativos gráficos, aparecieron aplicaciones que facilitaron la programación y la presentación de información en las pantallas de las computadoras personales. Primordialmente éstas computadoras utilizan el sistema operativo Windows, en cualquiera de sus modalidades, como podrían ser: 3.0, 3.11, 95, 98, 2000, Millenium, xp y sus respectivas variantes.

El entorno fue propicio para que surguieran herramientas de desarrollo gráfico, tales como: Visual Basic, Visual C, Delphi y otras más. Estas herramientas permiten realizar aplicaciones muy rápidamente, ya que están sustentados en la filosofía de orientación a objetos, que dicho sea de paso, presentan ciertos problemas en el mantenimiento de los sistemas desarrollados, ya que se pueden tener líneas de código en casi cualquier parte del programa, ya sea en la forma de inicio, en los botones, en controles, en librerías separadas, etc.

Así El Cliente, es una computadora personal con gran capacidad de procesamiento, con un ambiente muy agradable de desarrollo y de presentación de datos, conocido como GUI (*graphical user interface*), que ejecuta un programa que ha sido realizado aprovechando dicho ambiente gráfico de la máquina y tiene por objeto recopilar información, filtrarla y validarla con las reglas del propio negocio, identifica la acción a realizar sobre la información y finalmente envía una petición de atención al servidor en donde se encuentra centralizada la invaluable información.

Las aplicaciones estarán complementadas por API´s (*application program interface*) que permiten la conexión del programa residente en cliente, con las bases de datos existentes en el servidor o servidores Unix. Existen API´s tanto para las diferentes herramientas de desarrollo, como para los diversos manejadores de bases de datos conocidos.

En caso de contar con un sólo servidor Unix, estaremos hablando de una base de datos local, ya que podríamos distribuir la base (o las bases) de datos en diferentes servidores comunicados entre sí, generándose así una bases de datos distribuida.

Cada cliente utilizará sus propios recursos tanto para el desarrollo de la aplicación, como para *disparar* peticiones al servidor, mismas que podrían ser inserciones, consultas, actualizaciones y depuración de la información.

Al cliente también se le conoce comúnmente como el *front-end* de la arquitectura, ya que es quien en pocas palabras "*da la cara*" al usuario que utiliza la aplicación.

El Servidor

Es una computadora que deberá contar con gran capacidad de almacenamiento y gran velocidad de procesamiento, además de tener instalado alguno de los sabores de Unix, como sistema operativo.

En este equipo se tendrá instalado al menos una instancia de alguno de los principales manejadores de bases de datos relacionales, que se encargarán de las operaciones realizadas a cada una de las bases de datos que alojan.

El servidor recibirá las peticiones de servicio de los diferentes clientes a los que atiende, resolverá dicha petición y finalmente regresará la respuesta obtenida al cliente que lo solicita.

El desempeño de los manejadores de bases de datos y del propio sistema operativo, es fácilmente configurable con la ayuda de Linux, mismo que cuenta con un par de manejadores de bases de datos al ser instalado. Microsoft ha intentado atacar este mercado utilizando su sistema operativo Windows NT (*New Technology*), pero hasta el momento sigue presentando graves problemas en su uso y seguramente no lo podrá superar en breve.

Al servidor también se le conoce como el *back-end*, por estar "en la parte posterior" de la arquitectura.

La Red

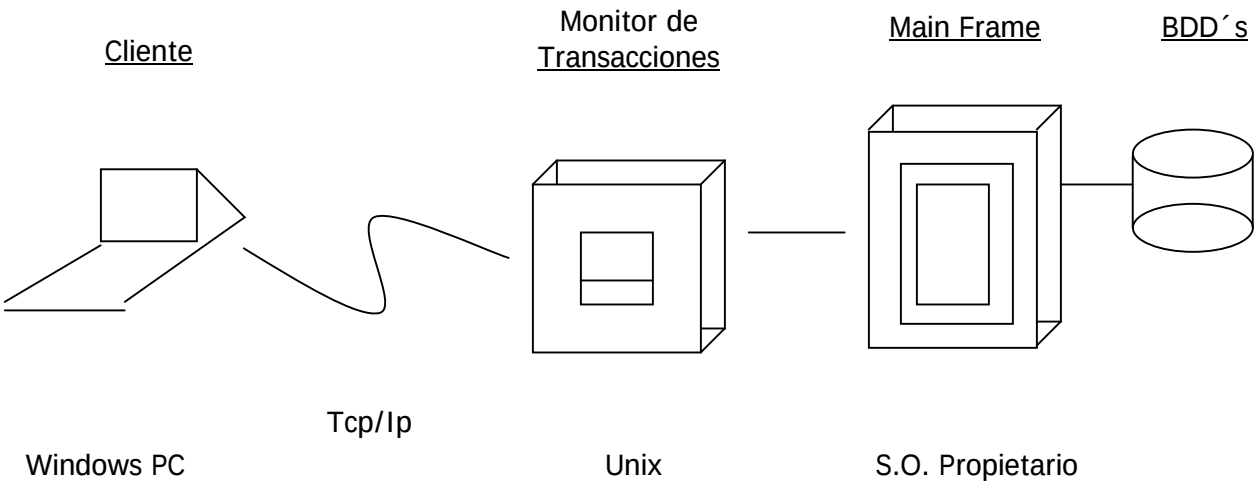
Toda la idea de poder repartir los diferentes procesos en diferentes computadoras, no podría ser sustentada sino no existiera una forma eficiente de poder conectar dichas computadores entre sí. La Red es precisamente quien permite esta comunicación entre diversas computadoras.

Para agilizar la transferencia de información, se han desarrollado diferentes protocolos de comunicación entre computadoras. De ellos, él que se ha mantenido y sigue evolucionando, es el protocolo TCP/IP (*Transport Control Protocol / Internet Protocol*), que en sí es un conjunto (*suite*) de protocolos que permiten diferentes formas de transferencia de información, entre los cuales se encuentra el http (HyperText *Transport Control*) que permite el manejo de hipertexto, tan utilizado en Internet.

La topología de la red tomo relevancia al percatarse que tanto debía viajar el mensaje enviado entre una computadora y otra, por lo que es muy variada y depende de la necesidad específica de cada caso a resolver.

b) De tres capas

La arquitectura cliente/servidor de tres capas, incluye un elemento más y está representada en el siguiente diagrama:



Aquí podemos apreciar que el cliente (front-end) reside en una computadora personal, al igual que en la recién explicada arquitectura cliente-servidor de dos capas.

El Servidor (back-end), son varios MainFrames con un sistema operativo propietario, en lugar de alguno de los sabores de Unix, y con un manejador de bases de datos también propietario. Los equipos utilizados principalmente son: IBM, UNISYS y TANDEM.

Hace su aparición un equipo más con sistema operativo Unix (o con alguno de sus sabores), que alberga el denominado monitor de transacciones. Este elemento es también conocido como el *middle-ware* y tiene como función primordial el poder encauzar las peticiones de los diferentes clientes hacia los diferentes servidores que albergan las bases de datos.

Los monitores de transacciones más comunes en el mercado, son *tuxedo* de BEA y *encina* de IBM. Dicho monitor de transacciones requiere de un programa cliente que debe estar presente en cada uno de los Main Frames, que albergan al manejador de bases de datos, para poder llevar a cabo la comunicación.

Entonces, las tres capas que forman esta arquitectura, tendrán diferentes equipos en cada una de ellas, dependiendo la función que deba desempeñar, la cuál puede ser:

- ✧ Front-end
  Que es el cliente que recibe, valida, envía la información capturada por el usuario y recibe la respuesta del servidor. Recordemos que generalmente utilizan alguna de las variantes de Windows como su sistema operativo, aprovechando alguna

de las computadoras personales que comparativamente son de menor costo.

- Middle-ware
  Que es un equipo con alguno de los sabores de Unix como sistema operativo y que tiene instalado algún de los monitores de transacciones que permitirán encontrar el back-end requerido por la petición del cliente.

- Back-end
  Que es uno o varios servidores, primordialmente main frames, con sistemas operativos propietarios, que albergan él o los manejadores de bases de datos con toda la invaluable información de la institución.

Todos éstas computadoras, obviamente, deben estar conectadas entre sí por una red (la cuál puede utilizar diferentes topologías dependiendo de la necesidad que se quiera cubrir), haciendo uso del protocolo de comunicación TCP/IP.

Características de Linux

Con la idea de hacer una lista completa de todas y cada una de las características que hacen de Linux un excelente sistema operativo, no podemos dejar de mencionar las siguientes:

- Es Multitarea, la palabra multitarea describe la habilidad de ejecutar varios programas al mismo tiempo. Linux utiliza la llamada multitarea preventiva, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.

- Es Multiusuario, es decir, muchos usuarios pueden hacer uso la misma computadora al mismo tiempo.

- Es Multiplataforma, las plataformas en las que en un principio se puede utilizar Linux son 386, 486. Pentium, Pentium Pro, Pentium II,Amiga y Atari, también existen versiones para su utilización en otras plataformas, como Alpha, ARM,MIPS, PowerPC y SPARC.

- Es Multiprocesador, que es el soporte para sistemas con más de un procesador. Esta disponible tanto para Intel, como para SPARC.

- Funciona en modo protegido 386.

- Cuenta con protección de la memoria entre procesos, de manera que uno de ellos no pueda colgar o pasmar el sistema.

- Realiza Carga de ejecutables por demanda, ya que sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.

- Política de copia en escritura para compartir páginas entre ejecutables, esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página (4Kb de memoria) se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.

- Memoria virtual usando paginación (sin intercambio de procesos completos) a disco, a una partición o un archivo en el sistema de archivos, o ambos, con la posibilidad de añadir más áreas de intercambio sobre la marcha. Un total de 16 zonas de intercambio de 128Mb de tamaño máximo pueden ser usadas en un momento dado con un límite teórico de 2Gb para intercambio. Este límite se puede aumentar fácilmente con el cambio de unas cuantas líneas en el código fuente.

- La memoria se gestiona como un recurso unificado para los programas de usuario y para el caché de disco, de tal forma que toda la memoria libre puede ser usada para caché y ésta puede a su vez ser reducida cuando se ejecuten grandes programas.

- Librerías compartidas de carga dinámica (DLL's) y librerías estáticas.

- Se realizan volcados o vaciados de estado (*core dumps*) para posibilitar los análisis post-mortem, permitiendo el uso de depuradores sobre los programas no sólo en ejecución sino también tras abortar éstos por cualquier motivo.

- Compatible con las versiones de Unix POSIX, System V y BSD a nivel fuente.

- Emulación de iBCS2, casi completamente compatible con SCO, SVR3 y SVR4 a nivel binario.

- Todo el código fuente está disponible, incluyendo el núcleo completo y todos los *drivers*, las herramientas de desarrollo y todos los programas de usuario; además todo ello se puede distribuir libremente. Hay algunos programas comerciales que están siendo ofrecidos para Linux actualmente sin código fuente, pero todo lo que ha sido gratuito sigue siendo gratuito.

- Control de tareas POSIX.

- Pseudo-terminales (pty's).

- Emulación de 387 en el núcleo, de tal forma que los programas no tengan que hacer su propia emulación matemática. Cualquier máquina que ejecute Linux parecerá dotada de coprocesador matemático. Por supuesto, si el ordenador ya tiene una FPU (unidad de coma flotante), esta será usada en lugar de la emulación, pudiendo incluso compilar tu propio kernel sin la emulación matemática y conseguir un pequeño ahorro de memoria.

- Soporte para muchos teclados nacionales o adaptados, y es bastante fácil añadir nuevos dinámicamente.

- Consolas virtuales múltiples, varias sesiones de *login* a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de video). Se crean dinámicamente y se pueden tener hasta 64.

- Soporte para varios sistemas de archivo comunes, incluyendo minix-1, Xenix y todos los sistemas de archivo típicos de System V, y tiene un avanzado sistema de archivos propio con una capacidad de hasta 4 Tb y nombres de archivos de hasta 255 caracteres de longitud.

- Acceso transparente a particiones MS-DOS (o a particiones OS/2 FAT) mediante un sistema de archivos especial: no es necesario ningún comando especial para usar la partición MS-DOS, esta parece un sistema de archivos normal de Unix (excepto por algunas restricciones en los nombres de archivo, y permisos). Las particiones comprimidas de MS-DOS 6 no son accesibles en este momento, y no se espera que lo sean en el futuro. El soporte para VFAT (WNT, Windows 95) ha sido añadido al núcleo de desarrollo y estará en la próxima versión estable.

- Un sistema de archivos especial llamado UMSDOS que permite que Linux sea instalado en un sistema de archivos DOS.

- Soporte en sólo lectura de HPFS-2 del OS/2 2.1

- Sistema de archivos de CD-ROM que lee todos los formatos estándar de CD-ROM.

- TCP/IP, incluyendo ftp, telnet, NFS, etc.

- Cuenta con Appletalk, para poder conectarse con equipos mac.

- Software cliente y servidor Netware.

- Lan Manager / Windows Native (SMB), software cliente y servidor.

- Diversos protocolos de red incluidos en el kernel: TCP, IPv4, IPv6, AX.25, X.25, IPX, DDP, Netrom, etc.

Linux frente a otros sistemas operativos

Linux es una muy buena alternativa frente a los demás sistemas operativos existentes actualmente en el mercado. Más allá de las ventajas evidentes de costo, ofrece características muy notables que lo hacen uno de los más robustos.

En comparación con las otras versiones de Unix para PC, la velocidad y confiabilidad de Linux son muy superiores. También está en ventaja sobre la disponibilidad de aplicaciones, ya que no hay mucha difusión de estos otros Unix (como son Solaris, XENIX o SCO) entre los usuarios de PC por sus altos costos.

Comparándolo con los diferentes sistemas operativos Microsoft Windows, Linux también sale ganando. Los bajos requisitos de hardware permiten hacer un sistema potente y útil de aquel 486 que algunos guardan en un armario. Esta misma característica permite aprovechar al máximo las capacidades de las computadoras más modernas que están en constante evolución.

Es poco práctico tener una PC con 16 Mb de RAM y ponerle un sistema operativo que ocupa 13 (que es lo que reporta sobre Windows 95 el *System Information de Symantec*). No solo es superior respecto a el sistema de multitarea y de administración de memoria, sino también en la capacidades de *networking* (conectividad a redes) y de multiusuario (aún comparando con sistemas multiusuario como NT).

Creemos que La única desventaja de Linux frente a estos sistemas, es la menor disponibilidad de software comercial, pero este problema disminuye con cada nuevo programa que se escribe para el proyecto GNU, ya que algunas empresas están desarrollando software comercial para Linux.

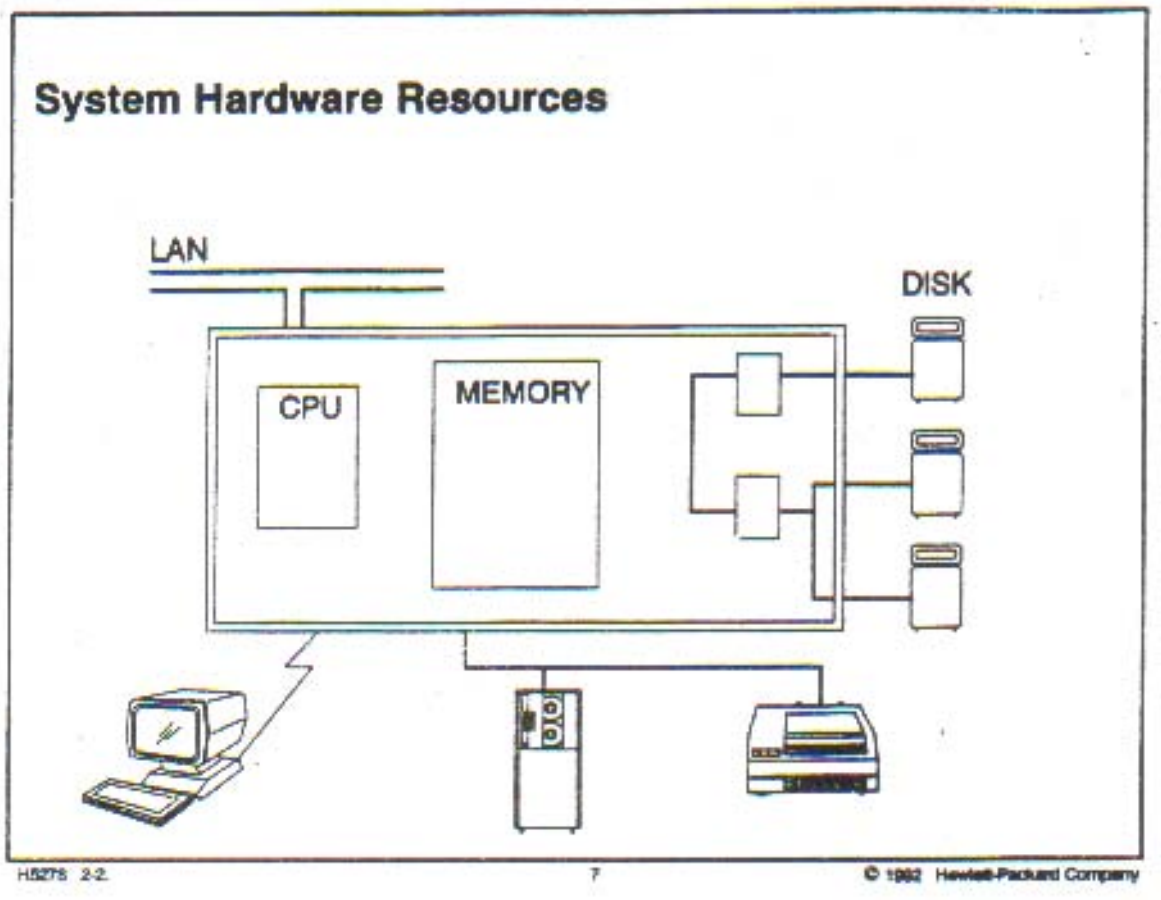
Todas éstas compañías de desarrollo de software han vislumbrado en éste sistema operativo un terreno fértil, en el cuál se pueden generar un sin fin de aplicaciones prácticas que lo fortalezcan aún más, como uno de los sistemas operativos más utilizados en el mundo de las computadoras personales.

Aprender sin pensar es inútil, pensar sin aprender es peligroso.
Confusio

6. Factores que afectan el rendimiento del Sistema Operativo

Como hemos comentado a lo largo del presente trabajo, el sistema operativo es el encargado del control de todos los dispositivos que lo forman, como son el procesador central que es donde se ejecutan las instrucciones, la memoria principal que es donde los programas y sus datos son cargados para su uso, los discos donde los programas y sus datos son almacenados cuando ya no son utilizados por el CPU, el teclado que permite ingresar información a la computadora, la pantalla utilizada para mostrar los datos ingresados y obtenidos de la computadora, así como impresoras, interfaces de red y otros dispositivos de entrada/salida.

Recursos del sistema



Así, en caso de que alguno de esos dispositivos presente problemas en su desempeño, repercutirá directamente en el rendimiento del propio sistema operativo. En sí, el

desempeño del sistema se refiere a que tan bien los recursos existentes en la computadora logran realizar el trabajo que están destinados a llevar a cabo.

Formas de medición

Para poder realizar una medición exacta del rendimiento de un sistema operativo, se han definido dos conceptos principales, los cuáles son:

- **Tiempo de respuesta del sistema**, que es el instante de tiempo entre que el usuario oprime la tecla *return* y recibe una respuesta de la computadora.

- **Throughput,** que es el número de transacciones logradas en un determinado lapso de tiempo.

Cualquiera de estas dos mediciones nos puede informar en que dispositivos se está invirtiendo más tiempo y, por ende, recursos del sistema. Aunque el *troughput* es mejor, ya que nos informa que tanto trabajo ha sido realmente realizado; sin embargo, el tiempo de respuesta es más tangible y por ende es más utilizado entre los administradores de sistemas.

En el ciclo de vida de un sistema el análisis de su desempeño juega un rol muy importante. Operar un sistema sin monitorearlo es como conducir un auto sin medir su consumo de gasolina, no sabemos cuando nos quedaremos sin combustible y quedemos estancados a un lado del camino.

La clave para el éxito en el manejo del desempeño es monitorear el sistema en forma regular, tomando decisiones acerca de la afinación del sistema o definir si se requieren de más recursos, antes de que el rendimiento del sistema se vuelva un problema grave que lleve a detener su funcionamiento.

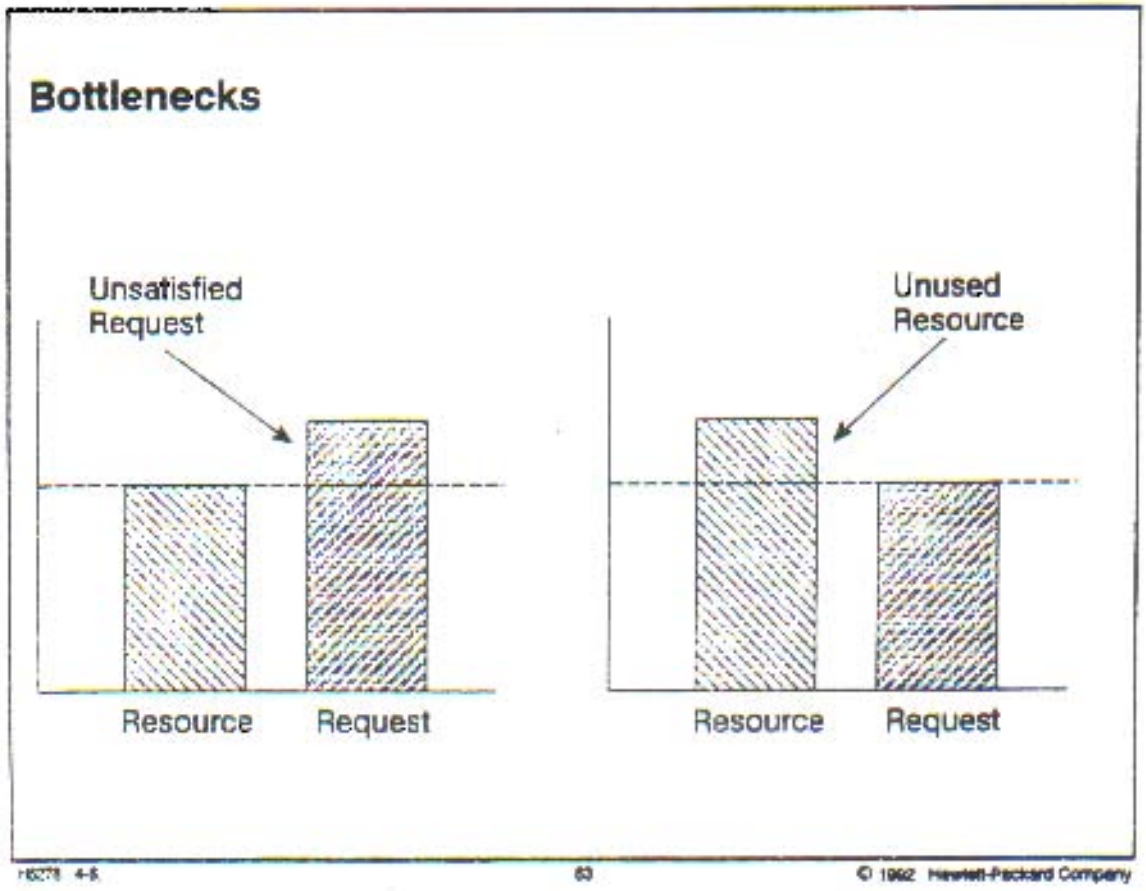
Las expectativas y las necesidades definen el buen o aceptable desempeño de un sistema. Por ello es esencial el definir correctamente estas expectativas.

Puesto que las necesidades varían de *site* en *site*, la habilidad para controlar el uso de los recursos del sistema y el tiempo de respuesta del mismo se vuelve un factor muy importante en el logro de la satisfacción del cliente y en la percepción de un buen desempeño del sistema.

Básicamente la operación del sistema debe mantener un alto *throughput* mediante la planeación eficaz de los procesos y el uso de los recursos compartidos.

Cuellos de Botella

Si el tamaño de una demanda excede el recurso disponible, este recurso se vuelve un cuello de botella. El recurso tiene más demanda de la que puede atender en ese momento.



En otras palabras, un cuello de botella es una limitación del desempeño del sistema debido a la insuficiencia de un componente de hardware o software.

Cuando el cuello de botella se presenta y es detectado oportunamente, podemos resolverlo de dos maneras diferentes:

- ✓ Reducir el tamaño de la demanda existente
- ✓ Incrementar el tamaño del recurso disponible

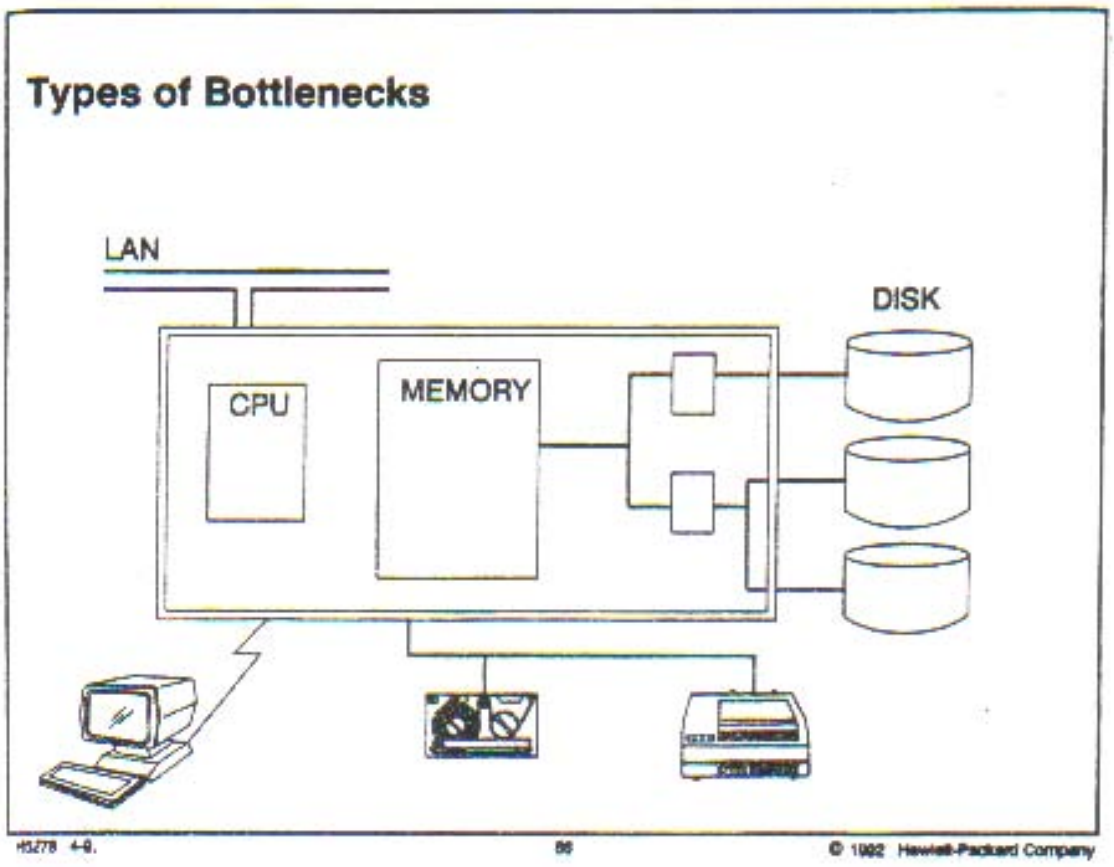
Generalmente, el aumentar el tamaño de algún recurso involucra la afinación del sistema o actualización del mismo. Por ello es de suma importancia identificar los cuellos de botella que puedan presentarse en el sistema. Cada uno de estos cuellos de botella pueden detectarse fácilmente identificando sus síntomas.

Es muy probable que un mismo síntoma pueda referirse a diferentes tipos de cuello de botella, por lo que es recomendable tomar todas las evidencias posibles para detectar efectivamente el problema y con ello definir claramente las acciones que deben realizarse para corregirlo.

Tipos de cuello de botella

Los diferentes tipos de cuellos de botella que pueden presentarse en el sistema operativo o equipo que estamos monitoreando, son:

- ✓ En el uso del procesador central
- ✓ En el uso de la Memoria
- ✓ En el uso del Disco duro
- ✓ En el uso de Otros dispositivos de entrada/ salida, como terminales, red, impresoras, etcétera



En si el cuello de botella puede presentarse en cualquiera de los recursos que componen nuestro sistema operativo.

Como comentamos inicialmente, probablemente los síntomas detectados puedan hacer referencia a diferentes tipos de cuellos de botella, por lo que debemos tomar todas las evidencias necesarias para poder identificar claramente el problema, y por ende, la solución más adecuada posible.

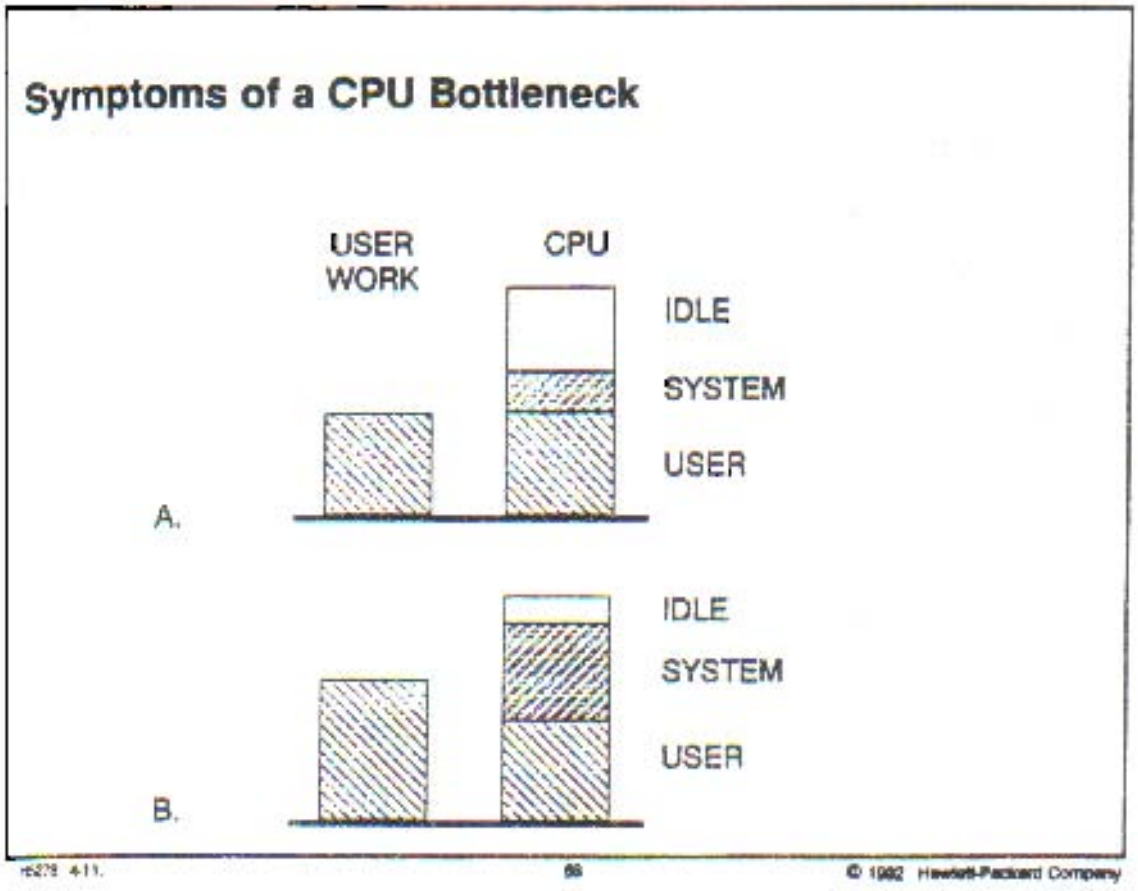
El cambiar o aumentar un recurso que se cree limitado en el sistema, no siempre es la solución más adecuada, ya que si no se ha identificado el problema podría solamente complicar aún más la situación.

Por eso consideramos de suma importancia el tomar un tiempo específico para monitorear el que hacer de nuestros sistemas de cómputo, definir previamente umbrales de desempeño y medir de la manera más precisa posible su desempeño.

Al mismo tiempo debemos especificar las acciones que deberán ser tomadas en caso de que los umbrales de desempeño óptimo sean alcanzados o superados.

Todas estas acciones deberán ser realizadas por el administrador del sistema, mismo que deberá sustentarlas fehacientemente.

a) Síntomas de cuellos de botella en el Procesador Central.



El uso más eficiente del CPU es el invertir todo el tiempo posible en ejecutar instrucciones que realicen trabajo para nosotros y no en ejecutar instrucciones de *overhead*. Si un gran porcentaje del tiempo es gastado en actividades de *overhead*, tendremos un uso ineficiente del recurso.

Podemos identificar la existencia de un cuello de botella en el CPU si el sistema muestra algunas de las siguientes actividades:

- ✓ No exista tiempo ocioso en el procesador
- ✓ Tener una larga cola de procesos que esperan ser ejecutados
- ✓ Alta actividad en el modo de usuario
- ✓ Alta actividad en modo sistema

Una vez detectado un cuello de botella en el CPU, debemos realizar las acciones recomendadas para erradicarlo, mismas que se encuentran más adelante en este mismo capítulo.

b) Síntomas de cuellos de botella en la Memoria.

La memoria principal del sistema está dividida en dos partes, las cuáles son: dinámica y reservada. La memoria dinámica es utilizada para la ejecución de procesos, almacenar la pila de datos y manejar segmentos de memoria compartida; mientras que la memoria reservada es utilizada por las tablas del sistema, algunas estructuras de datos y el uso de *buffers*. El uso ineficiente del recurso puede ocasionar un cuello de botella.

Para poder identificar la existencia de un cuello de botella en la memora principal del sistema, debemos atender a los siguientes síntomas:

- ✓ Alta actividad de alternancia (*swapeo*)
- ✓ Alta actividad de paginación
- ✓ Memoria libre muy limitada con una gran actividad de la memoria virtual
- ✓ Alta actividad del disco en un dispositivo de *swapeo*
- ✓ Alto uso del procesador central en el modo sistema, debido al *swapeo* y paginación
- ✓ Una larga cola de procesos con poco tiempo ocioso

Una vez que hemos detectado la existencia de un cuello de botella en la memoria principal, debemos realizar las acciones recomendadas para erradicarlo, mismas que se encuentran más adelante en este mismo capítulo.

c) Síntomas de cuellos de botella en el disco.

El disco o memoria de almacenamiento del sistema, presenta demandas por algunas de los siguientes factores:

- ✧ Programas de usuarios
- ✧ Programas del sistema
- ✧ Espacio destinado a *swap*

Los archivos temporales son utilizados tanto por programas del sistema como por programas de aplicación (como son los creados en el directorio /tmp cuando se hace uso del editor *vi*). Generalmente la creación y uso de estos archivos es transparente y no nos percatamos de su existencia. No obstante, pueden generar una actividad muy pesada en el disco mientras el programa se encuentra en ejecución.

Podemos fácilmente identificar la existencia de un cuello de botella en el disco si se presentan algunos de los siguientes síntomas:

- ✓ Alta actividad en el disco
- ✓ CPU ocioso esperando por entradas / salidas
- ✓ Alto porcentaje de lecturas y escrituras físicas (%rcache, %wcache)
- ✓ Una larga cola de procesos en espera, teniendo tiempo ocioso en el CPU

Una vez detectado un cuello de botella en el disco, debemos realizar las acciones recomendadas para erradicarlo, mismas que se encuentran más adelante en este mismo capítulo.

d) Otros síntomas de cuellos de botella en entrada/salida.

Otros dispositivos de entrada/salida pueden ser potencialmente cuellos de botella, aunque es menos común que los cuellos de botella que se presentan en el disco. Podrían presentarse con algunas aplicaciones o en cierto momento durante el día.

Por ejemplo, una aplicación puede imprimir un reporte extenso muy rápidamente, pero tal vez requiera de mucho tiempo para hacerlo; en este caso la velocidad de la impresora es el cuello de botella para la aplicación.

Existen dos elementos principales a considerar por cada una de las unidades de entrada/salida. Una es el aspecto del hardware o la conexión del bus; la otra es el *overhead* del sistema relacionado con el controlador del dispositivo de entrada/salida especificado.

Para poder identificar la existencia de un cuello de botella en algún otro dispositivo de entrada/salida, debemos estar atentos a los siguientes síntomas:

- ✓ Alta actividad de entrada/salida
- ✓ Pérdida de datos
- ✓ Inaceptable *throughput*

Una vez que hemos detectado la existencia de un cuello de botella en algún dispositivo de entrada/salida, debemos realizar las acciones recomendadas para erradicarlo, mismas que se encuentran más adelante en este mismo capítulo.

e) Síntomas de cuellos de botella en los recursos del Kernel.

Los recursos del Kernel del sistema operativo, pueden presentar cuellos de botella primordialmente por el uso inadecuado de:

- ✧ Semáforos
- ✧ Colas de mensajes
- ✧ Memoria compartida
- ✧ Tablas del sistema

Los parámetros definidos para el funcionamiento de los recursos del kernel, se encuentran almacenados en el archivo: /etc/sysdef.

Entre los procesos del sistema operativo que consumen una gran cantidad de los recursos del sistema, se encuentran:

- ✓ Bases de datos relacionales
- ✓ Algunos procesos del sistema
- ✓ Aplicaciones existentes

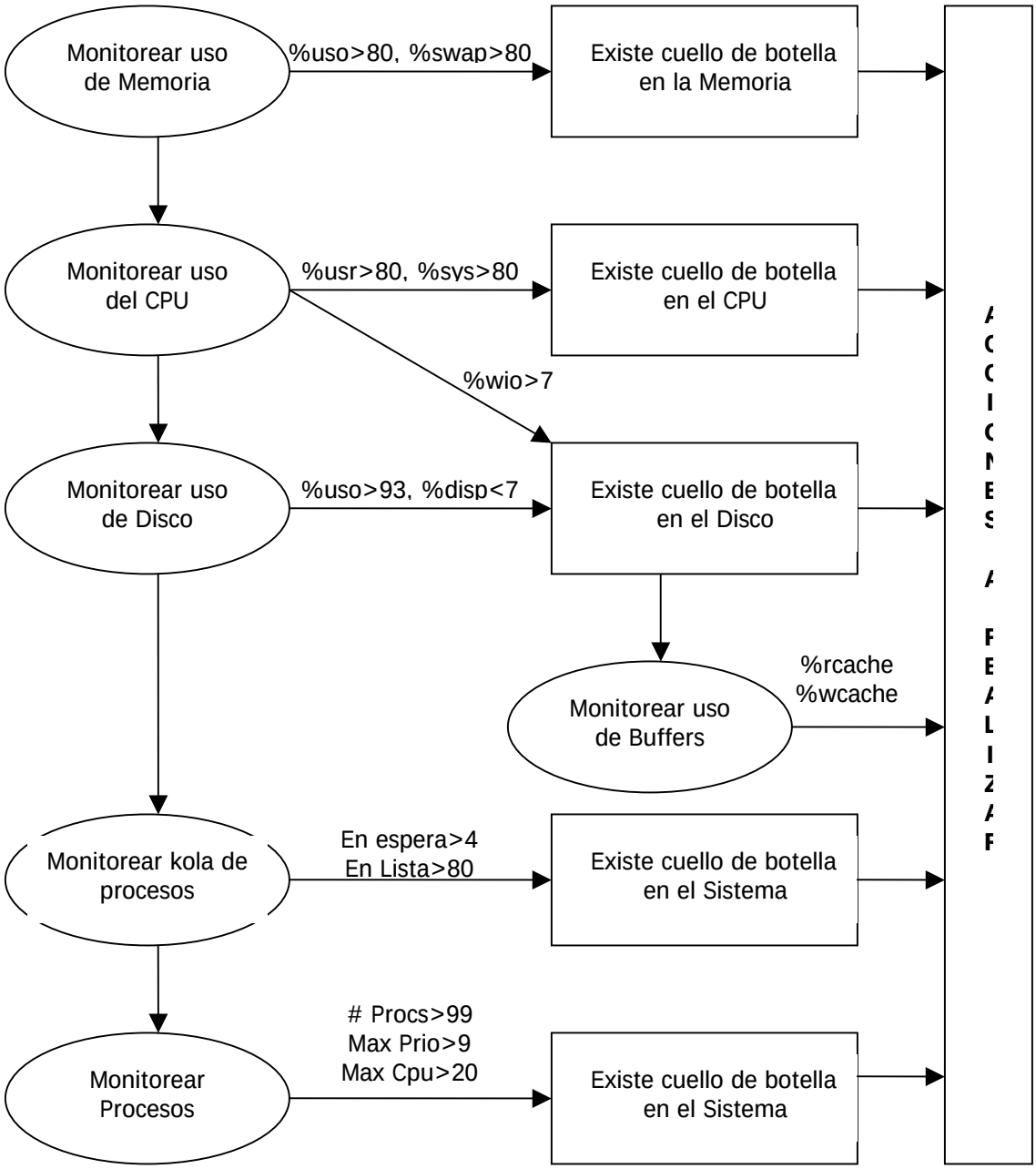
Los cuellos de botella del *Kernel* incluyen el eficiente manejo tanto de colas de mensajes como de semáforos. Los semáforos son esencialmente utilizados por las bases de datos relacionales.

Fácilmente podemos identificar la existencia de un cuello de botella en los recursos del kernel, si se presentan algunos de los siguientes síntomas:

- ✓ Alto uso del CPU por el sistema
- ✓ Alto promedio de llamadas el sistema
- ✓ Mensajes de error recibidos por las llamadas al sistema
- ✓ Limitación del sistema por el CPU

Diagnostico de cuellos de botella.

Para poder identificar si existen o no cuellos de botella en los recursos de nuestro sistema Linux debemos seguir el siguiente diagrama de procesos:



Revisando los procesos existentes en el sistema e identificando la actividad de "*swapeo*" a disco podemos identificar si existe o no una limitación en el sistema operativo por el mal uso de la memoria.

Si el sistema no esta siendo limitado por la memoria, debemos revisar si no existen limitaciones por uso del disco. Si se tienen discos con actividad pesada, muy probablemente el sistema esta siendo limitado por el mal uso del disco.

Debemos examinar la razón por la cuál el sistema está limitado por disco, revisando el porcentaje de lecturas y escrituras al buffer. Podemos también revisar el tipo de acceso a archivos que se está utilizando.

Finalmente, debemos revisar los procesos existentes para así determinar si existe algún cuello de botella en el uso del CPU. Si el porcentaje de tiempo ocioso del CPU en la espera de bloques de entrada/salida es alto, debemos verificarlo fehacientemente para evitar así un cuello de botella en disco.

En el caso de que no exista un tiempo ocioso para el sistema y que el porcentaje de tiempo en el modo usuario es alto, tendremos entonces una limitación por CPU.

Metas de la afinación del desempeño.

La meta a alcanzar al poner a punto el sistema, es establecer una base de desempeño que pueda hacer posible ejercer mayor control sobre el uso del sistema y estar preparado para cambios repentinos a ese uso. La base de desempeño consiste en definir rangos aceptables de funcionamiento de cada uno de los recursos del sistema y clarificar umbrales máximos de acción, los cuáles no deben ser alcanzados.

Cuando los umbrales definidos para cada uno de los recursos son alcanzados o superados, debemos tener claramente definidas las acciones a realizar para normalizar su funcionamiento.

La finalidad que perseguimos con la afinación del desempeño del sistema, es:

- Maximizar el *throughput* existente
- Mantener un tiempo de respuesta aceptable
- Utilizar eficientemente los recursos del sistema

Podemos apreciar que la principal meta en la afinación de un sistema es la de maximizar el *throughput* (procesos realizados en el sistema por minuto), mientras se mantiene un

tiempo de respuesta aceptable. Así la afinación nos permitirá hacer uso de los recursos del sistema de la mejor manera posible.

a) Afinación del desempeño del CPU

Para poner a punto el uso del procesador central del sistema, debemos realizar las siguientes acciones con el fin de mejorar su desempeño:

- ✓ Balancear las cargas de trabajo
- ✓ Eficientar el tiempo dedicado a control de procesos
- ✓ Realizar una actualización de hardware

Las acciones propuestas consisten en verificar el porcentaje del uso del procesador dedicado a aplicaciones propias del sistema, a aplicaciones de los diferentes usuarios y a tiempo inactivo u ocioso.

Cuando ninguna de las acciones correctivas pueden solucionar el problema detectado, y solamente entonces, no quedará más remedio que actualizar el procesador central de nuestro equipo.

b) Afinación del desempeño de la Memoria

Para poner a punto el uso de la memoria principal del sistema, debemos realizar las siguientes acciones con el fin de mejorar su desempeño:

- ✓ Manejar la memoria más eficazmente
- ✓ Afinar las aplicaciones del usuario para eficientar el espacio disponible
- ✓ Agregar más memoria al sistema

Las acciones propuestas tienen como fin el verificar el porcentaje de uso tanto de la memoria principal como el dedicado al área *swap* (recordemos que es un espacio de memoria principal generado en el disco duro), para que se conserven en el rango de funcionamiento definido.

En caso de que ninguna de las acciones correctivas permitan solucionar el problema detectado, deberemos entonces y sólo entonces, agregar más memoria a nuestro equipo de cómputo. En este extremo caso, deberemos detectar que tipo de memoria principal utiliza el sistema, ya que los modelos cambian muy rápido y generalmente se instalan por

pares. Debemos tener mucho cuidado en instalar el mismo modelo y sobretodo conseguir que trabajen a la misma velocidad.

c) Afinación del desempeño del Disco

Para poner a punto el uso del disco del sistema, debemos realizar las siguientes acciones con el fin de mejorar su desempeño:

- ✓ Balancear la carga del disco
- ✓ Poner a punto el sistema de archivos
- ✓ Reservar espacio para archivos
- ✓ Hacer uso de *raw devices*
- ✓ Revisar la forma en que las aplicaciones utilizan el disco
- ✓ Permitir que los diferentes periféricos existentes compitan por separado
- ✓ Agregar disco

Las acciones recomendadas tienen por objeto verificar tanto el porcentaje del uso del disco, como el número de transacciones generadas por segundo hacia el mismo, constatando que se mantenga en el rango definido para su óptimo funcionamiento, sin alcanzar los críticos umbrales.

Cuando ninguna de las acciones correctivas pueden solucionar el problema detectado, y solamente entonces, no quedará más remedio que ampliar o cambiar el disco duro de nuestro equipo. En ese crítico caso deberemos también considerar la forma en que se realizará la migración de todas y cada una de las aplicaciones que se tienen en ejecución en el sistema.

d) Afinación del desempeño de entrada/salida

Para poner a punto el uso de entrada/salida del sistema, debemos realizar las siguientes acciones con el fin de mejorar su desempeño:

- ✓ Retirar las tarjetas multiplexor que no se estén ocupando
- ✓ Hacer uso de las tarjetas de interfaz adecuadas
- ✓ Separar periféricos competitivos

- ✓ Utilizar tarjetas RTI (*Real Time Interface*) para las aplicaciones de tiempo real

e) Afinación del desempeño del kernel

Las principales acciones que deben ser tomadas al presentarse cuellos de botella en el *kernel*, son:

- ✓ Revisar porcentaje de uso del CPU por el sistema
- ✓ Analizar las llamadas el sistema
- ✓ Cotejar los mensajes de error de las aplicaciones
- ✓ Verificar la información del desborde de tablas (*overflow*)
- ✓ Tomar en cuenta la información de asignación de IPC

Metodología para la afinación.

La metodología de la afinación del desempeño del sistema, es un plan de acciones que conduce a:

- ✓ Definir los objetivos del desempeño
- ✓ Evaluar el desempeño actual
- ✓ Identificar cuellos de botella
- ✓ Determinar acciones a realizar

El propósito primordial de la metodología de afinación, es el definir una base de la cuál fincaremos acciones a realizar; así que debemos primeramente dejar claro el rango de funcionamiento de cada uno de los recursos que componen el sistema, al cuál hemos conocido a lo largo de éste capítulo como su óptimo desempeño.

Después debemos monitorear regularmente su funcionamiento y sobretodo debemos evaluar cabalmente el desempeño del sistema, con el firme propósito de poder detectar posibles cuellos de botella que afecten el correcto rendimiento del equipo, basándonos tanto en el *throughput* y en el tiempo de respuesta existente, tal y como lo mencionamos al inicio de este capítulo.

Finalmente debemos dedicarnos a la interesante tarea de realizar el *action check list* o lista de acciones que deberán ser realizadas, con el único propósito de mejorar el desempeño del o de los recursos del sistema que están presentando problemas en su rendimiento y que de no hacerlo oportunamente podrían presentarse situaciones irreversibles.

Nunca hay viento favorable para el que no sabe hacia donde va.
Séneca

7. Implementación de un sistema que realice el análisis y afinación del rendimiento de Linux

El monitorear o supervisar el quehacer de un sistema operativo multiusuario y multitarea como lo es Linux, requiere primordialmente el definir claramente lo que se desea supervisar, para así poder tomar acciones que permitan corregir el o los problemas que se pudieran presentar.

Este monitoreo consiste en verificar el uso de los recursos existentes en el equipo, entre los cuáles se encuentran:

1. Uso del procesador central
2. Uso de la memoria principal
3. Uso del disco o memoria secundaria
4. Uso de buffers
5. Estado actual de las colas de procesos
6. Estado actual de los procesos

Con el firme propósito de poder mejorar el desempeño o rendimiento del equipo, y con ello, adecuar el tiempo de respuesta que el sistema está presentando para cada uno de sus usuarios.

Para implementar un sistema que pueda realizar el análisis y posteriormente la afinación de los recursos utilizados por el sistema operativo Linux, decidimos generar un conjunto de directorios utilizando la clave de administración del equipo (root), para así facilitar la creación del sistema de monitoreo y afinación.

Utilizando la clave de administración, cuyo directorio tiene la ruta: /root, generamos una estructura de directorios que permitiera el fácil desarrollo y mantenimiento del sistema de monitoreo.

Estructura de directorios

Los directorios creados bajo la ruta de la clave de administración root, son:

/root/bins
/root/bitacoras
/root/comandos
/root/cprogs
/root/make
/root/monitor
/root/scripts y
/root/textos.

Estos directorios fueron creados para albergar la siguiente información:

/bins
Es el directorio de programas binarios.
Es donde se encuentran todos los programas ejecutables utilizados por el sistema. Hacemos la emulación del directorio bin de Linux.

/bitacoras
Es el directorio de bitácoras de eventos.
Cada vez que el sistema detecte que el sistema supere los umbrales de funcionamiento óptimos, se creará una bitácora indicando que umbral ha sido superado. El nombre de la bitácora estará formado de la siguiente manera: *nnmmmhh*, en donde *nn* será el día de creación, *mmm* es el nombre del mes y *hh* indica tanto la hora como el minuto en el que fue creada.

/comandos
Es el directorio de comandos escritos en lenguaje shell.
Aquí se localizan los comandos shell que permiten la navegación y control de la estructura de directorios creada.

/cprogs
Es el directorio de programas fuente escritos en lenguaje C.
Son rutinas que permiten dar una mejor apariencia al sistema al ser invocados desde rutinas shell, que son ejecutados en modo carácter.
En este directorio también se tienen programas que son utilizados como librerías.

/make
Es el directorio en donde se encuentra el archivo *makefile* y el cuál define la forma en que deben compilarse los diferentes programas que dan forma al sistema de monitoreo y afinación de Linux.

/monitor
Es el directorio donde reside propiamente el sistema de monitoreo y afinación del sistema operativo Linux, al cuál hemos denominado myalix.

/scripts
Es el directorio de scripts realizados en programación shell y que son de propósito general.

/textos
Directorio que contiene el manual de usuario de myalix y el cual puede ser invocado desde la línea de comandos mediante: ayuda myalix.

El shell utilizado en la sesión es *Bourn Again Shell*, mejor conocido como bash y que es una nueva generación del legendario *bourn shell* utilizado desde las primeras versiones Unix. Para poder hacer uso de los diferentes directorios creados en la clave, se definieron rutas de búsqueda de comandos y funciones para la sesión, en el archivo que Linux redhat

utiliza para iniciar la sesión en modo carácter, el cuál es: .bash_profile. Cabe aclarar que en el antiguo bourne shell, el archivo de inicialización era el .profile. Puede observarse que ambos archivos inician con un punto para que puedan esconderse en la sesión.

Algunas líneas de dicho archivo son:

```
PATH=$PATH:$HOME/bin
MYPATH=$HOME/monitor; $HOME/bins; $HOME/comandos
PATH=$PATH:$MYPATH:.
export PATH MYPATH
        .
        .
        .

root()
{
    . root
}

bins()
{
    . bins
}

mya()
{
    . mya
}
        .
        .
        .
```

En la primera parte del archivo .bash_profile, derfinimos la ruta de búsqueda de comandos, denominada MYPATH, la cuál consta de los programas ejecutables almacenados en el directorio bins (ejecutables) y de los programas shell localizados ya sea en el directorio de comandos o en el directorio del sistema de monitoreo.

En la segunda parte, se declararon tantas funciones como directorios creados, ya que es necesario definir funciones con el mismo nombre del comando shell para que al invocarlo realice los cambios estipulados en cada uno de dichos comandos y sobre todo tengan efecto en la sesión.

Estos comandos son utilizados para facilitar la navegación dentro de la estructura de directorios creada en la clave de administración del sistema.

Las acciones principales que realizan los comandos shell utilizados para la navegación en la estructura, es el cambio de directorio actual, el cambio de prompt que identifica a cada

uno de los directorios y la ejecución de diferentes comandos Linux, dependiendo de cada uno de dichos comandos.

La estructura general de comandos de navegación está formado por:

```
PS1= “Comandos ^[[5m>^ [[0m ”
NEWDIR= < ruta del directorio >

cd $NEWDIR
  ... comandos Linux ...
```

En la variable de ambiente PS1 utilizada para definir el prompt de la sesión, incluímos secuencias de escape que permiten activar ciertas acciones gráficas como son el uso de video inverso, el flasheo o el bold, permitiendo de esta forma dar ciertas características gráficas a un ambiente completamente carácter.

Al iniciarse la sesión en ambiente carácter con la clave root, el archivo .bash_profile ejecutará una rutina de seguridad, misma que presentará un mensaje en la pantalla informando al usuario que está haciendo uso de la clave privilegiada del sistema, por lo que le pedirá una contraseña de inicio de sesión.

El usuario contará con tres oportunidades de escribir correctamente la contraseña de inicio, de lo contrario la sesión se cerrará y se le dará aviso al usuario.

La rutina de seguridad, denominada ChecaClave, genera un código de retorno dependiendo si se dio o no correctamente la contraseña de entrada, mismo que recibe shell y toma diferentes acciones dependiendo del valor obtenido.

Contraseña de inicio

El programa shell que realiza la verificación de la contraseña de inició de sesión, es el siguiente:

```
#############################################################
# Programa: ChecaClave
# Objetivo  : Verificar la clave de inicio de sesión
# Lenguaje: shell
#
# Autor: Miguel Angel Herrera MC
# Fecha de creación: mayo del 2004    Ultima modificación: mayo del 2004
#################################################<mahmc>###

#
# Variables
#
INTENTOS=0
TOTAL_INTENTOS=3
CONTINUA=true
USUARIOS= $( who ¡ wc –l )
```

```
#
# Programa Principal
#
clear
tput cup 12 20
echo –e “\aEstas hacienda uso de la clave de administrador ...\c”
sleep 3
tput cup 12 20
echo –e “                                                    ”
while $CONTINUA
do
   tput cup 12 30
   echo –e “Contraseña: *******\b\b\b\c”
   read –s PASSIN
   if [ “$PASSIN” = “sea” ] ; then
       CONTINUA=false
       tput clear
       tput cup 12 33
       echo “Bienvenido !!”
       sleep 1
       return 1
   else
      let INTENTOS=$INTENTOS + 1
       tput cup 12 33
       echo “Clave Incorrecta.”
       sleep 1
      if  [ $INTENTOS –eq $TOTAL_INTENTOS ] ; then
            tput clear
            tput cup 12 31
            echo “Hasta Pronto !!”
            sleep 1
            kill -9 0
      fi
   fi
done
sleep 3
clear
date +”%t%t%t%t%t%t%t%a %d/%b %H:%M”
cal
echo –e “\n”
echo –e “Usuarios en sesion: ${USUARIOS}”
who
echo –e “\n”
ls
echo –e “\n”
#################################################<mahmc>###
```

La rutina pide una contraseña de entrada y el usuario cuenta con tres oportunidades para digitarla correctamente, si en estas oportunidades el usuario no proporciona correctamente dicha contraseña, la sesión será cerrada por el sistema operativo enviando un mensaje de despedida.

Una vez dentro de la clave de administración y al tener activo los comandos de navegación, fácilmente podremos ingresar al directorio de desarrollo del sistema de Monitoreo y Afinación, mediante el comando: *mya*.

Para invocar en cualquier momento el Sistema de Monitoreo y Análisis del Sistema Operativo Linux, no es necesario ingresar al directorio de desarrollo, bastará con invocarlo con el comando: *myalix*.

myalix

El sistema de monitoreo y afinación del sistema operativo Linux (myalix) está desarrollado en programación shell para poder interactuar directamente con los recursos del sistema operativo y se complementa con llamadas a programas escritos en lenguaje C, que serán los encargados de dar una mejor apariencia a los diagnósticos realizados en el equipo.

Myalix es el módulo encargado de mostrar y controla el menú principal del sistema, para así invocar a cada uno de los módulos que lo componen y que tienen por fin el monitorear un determinado recurso del sistema operativo y dar acciones a realizar con el fin de poder afinarlo en la mejor forma posible.

Al invocar el sistema myalix, el programa divide la pantalla en tres diferentes secciones que permitirán identificar claramente el desempeño del equipo.

La primera sección de la pantalla es utilizada para presentar los menús de acción y está localizada en las tres primeras líneas de la pantalla. Aquí se presentan los diferentes menús, sus opciones, la fecha y hora de la muestra.

Así, al iniciar, en la primera línea se pregunta por que factor se quiere monitorear y las opciones válidas a utilizar se presentan en la segunda línea, las cuáles pueden ser:

1) Cpu
2) Memoria
3) Disco
4) Buffers
5) Kolas
6) Procesos y
7) Salir

La selección se realiza por la letra mayúscula y las leyendas del área de menús cambian dependiendo de la opción deseada. La segunda área de pantalla, denominada de despliegue, comprende de la cuarta línea a la veintitrés y es el área utilizada para presentará las estadísticas del sistema.

Ahí mismo se presentarán mensajes que informarán sobre la forma de realizar la afinación si es que alguno de los monitores detecto que los umbrales de referencia han sido alcanzados o superados.

La tercera sección de pantalla, denominada de mensajes, está formada por las dos últimas líneas de la pantalla y se utiliza para mostrar el tiempo en el que el sistema ha estado en servicio, el número de usuarios conectados, despliega los umbrales de funcionamiento del recurso monitoreado, permite enviar mensajes al usuario y mantiene siempre a la vista el nombre del sistema que hemos desarrollado.

Al iniciar el sistema myalix, se presentan en el área de despliegue la actividad de los usuarios conectados al sistema, lo que están ejecutando en ese momento, presentando siempre fecha y hora de la muestra tomada, así como el tiempo que tiene activo el equipo.

El programa shell myalix, es el menú principal que permite definir las áreas dentro de la pantalla: menús, despliegue y mensajes, además de tener el control de seis diferentes módulos, los cuáles están realizados también en programación shell con la idea de estructurar bien el sistema, facilitando así su desarrollo y posterior mantenimiento.

Cada uno de estos módulos monitoreará y recomendará acciones a realizar con el fin de poder afinar, si es posible, el cuello de botella detectado en alguno de los elementos que pudieran afectar el rendimiento del sistema. Toda la programación hace uso de comandos propios de Linux, mostrados en el anexo de éste trabajo, haciendo un total de más de tres mil líneas.

El programa myalix completo, que invoca a los módulos de monitoreo y afinación, lo mostramos a continuación:

```
############################################################
# Programa: myalix
# Objetivo: Monitorear y Afinar los recursos del Sistema Operativo Linux.
#             Las opciones validas del menu presentado son:
#
#            C - Monitoreo del cpu
#            M - Monitoreo de la memoria
#            D - Monitoreo del disco
#            B - Monitoreo de buffers
#            K - Monitoreo del colas de procesos
#            P - Monitoreo de procesos activos
#            S - salir
#
#         Al iniciar se muestran los  veinte  procesos  mas  antiguos  que  se
#         estan aun ejecutando. Cuando  los  umbrales  de  funcionamiento  son
#         alcanzados o superados se genera una bitacora de eventos registrando
#         la situacion del equipo.
#
# Lenguaje: shell
# Autor   : Miguel Angel Herrera MC
# Fecha de Creacion: Mayo del 2004                    Ultima modificacion: Ago del 2004
#############################################################mahmc###

#
# Constantes
#
```

```
B="`tput smso`"
N="`tput sgr0`"

#
# Variables
#

ContinuaMain=true
SleepTime=2
CONTINUAMON=true
CONTINUAAFI=true
DEFSALIR=3
DEFREGRESAR=9
MUESTRAS=5

#
# Creacion de la bitacora de eventos
#

FechaAct=`date +"%d%b%H%M"`
echo $FechaAct > /root/bitacoras/TodayFile

#
# Funciones
#

DespliegaPresentacion()
{
 clear
 echo -e "\n\n\n\n\n\n"
 echo -e "\a"
 echo -e "\t\t MM    MM  Y    Y   A      LL       III  X    X"
 echo -e "\t\t M M M M   Y Y    A A     LL       III   X X "
 echo -e "\t\t M   M  M    Y   AAAAA    LL       III    X  "
 echo -e "\t\t M     M   Y   A     A   LL       III   X X "
 echo -e "\t\t M     M  Y   A       A  LLLLLLL  III  X    X"
 echo -e "\n\t\t${B}        Monitoreo y Afinacion de Linux       ${N}"
 echo -e "\n"
 echo -e "\t\t\t\t\t\t\tmahmc\b\b\b\b\b\c"
 sleep 5
 read algo
 clear
} #end DespliegaPresentacion

EsperaEnter()
{
 tput cup 24 48
 echo -e "\t\t\t\t\t${B} Oprime <return> para continuar ...${N}\c"
 read ANSW
} #EsperaEnter
```

```
Espacios()
{
# Parametros de entrada
Linea=""
Espacio=" "
NumEsp=$1

let CuentaEspacios=0
while [ "$CuentaEspacios" -le "$NumEsp" ]
do
   Linea=$Linea$Espacio
   let CuentaEspacios=CuentaEspacios+1
done
#echo $Linea
} #end Espacios

Marco()
{
# Parametros de entrada
let Ren1=$1 ; let Con1=$Ren1
let Col1=$2 ; let Con2=$Col1
let Ren2=$3
let Col2=$4
let TipoCuadro=$5

# Constantes
B="`tput smso`"
N="`tput sgr0`"

# Variables
Cadena=" "
Long=0

# Main Function de Marco
clear
let Long=$Col2-$Col1
tput cup 2 2 ; echo $Ren1, $Col1, $Ren2, $Col2, $Con1, $Con2, $Long

while [ "$Con1" -le "$Ren2" ]
do
#   while [ "$Con2" -le "$Col2" ]
#   do
#      echo $Con1 $Con2
#      tput cup $Con1 $Con2 ; echo "${B} ${N}"
      Cadena="                                                                          "
      Cadena2="                                                                         "
      Cadena3="+-------------------------------------------------------------------------+"
#      espacios 80
#      Cadena=$Linea

      tput cup $Con1
      case $TipoCuadro in
          1) echo "$Cadena"
```

```
          ;;
        2) if [ $Con1 = $Ren1 -o $Con1 = $Ren2  ] ; then
                echo "${B}$Cadena${N}"
           else
                echo "${B} ${N}$Cadena2${B} ${N}"
           fi
           ;;
          3) echo "${B}$Cadena${N}"
          ;;
        4) if [ $Con1 = $Ren1 -o $Con1 = $Ren2  ] ; then
                echo "$Cadena3"
           else
                echo "|$Cadena2|"
           fi
           ;;
       esac
#       let Con2=Con2+1
#    done
    let Con1=Con1+1
    let Con2=$Col1
done
} #end Marco

#
# Programa Principal
#

DespliegaPresentacion
while $ContinuaMain
do
  MenuIni
  Usuarios=`who | wc -l `
  let Usuarios=Usuarios+10
  let Usuarios=Usuarios-10
# tput cup 1 60
# echo "${Usuarios} Usuarios"
  tput cup 2 60
  echo "${B}`date +"%a %d/%b  %H:%M:%S"`${N}"
  tput cup 23 0
  echo "${B} Up`uptime|cut -c14-19`, ${Usuarios} Usu ${N}"
  tput cup 3 1
# sar -r | head
# ps -eo "%p   %c   %t   %U   %y   %x" | head -20
  w > UsuIni
  lineasIni=`more UsuIni | wc -l`
  let lineasIni=lineasIni-1
  tail -$lineasIni UsuIni

  tput cup 0 13
  echo -e "\b\c"
  read OpcMain

#  Opcion por Omision
```

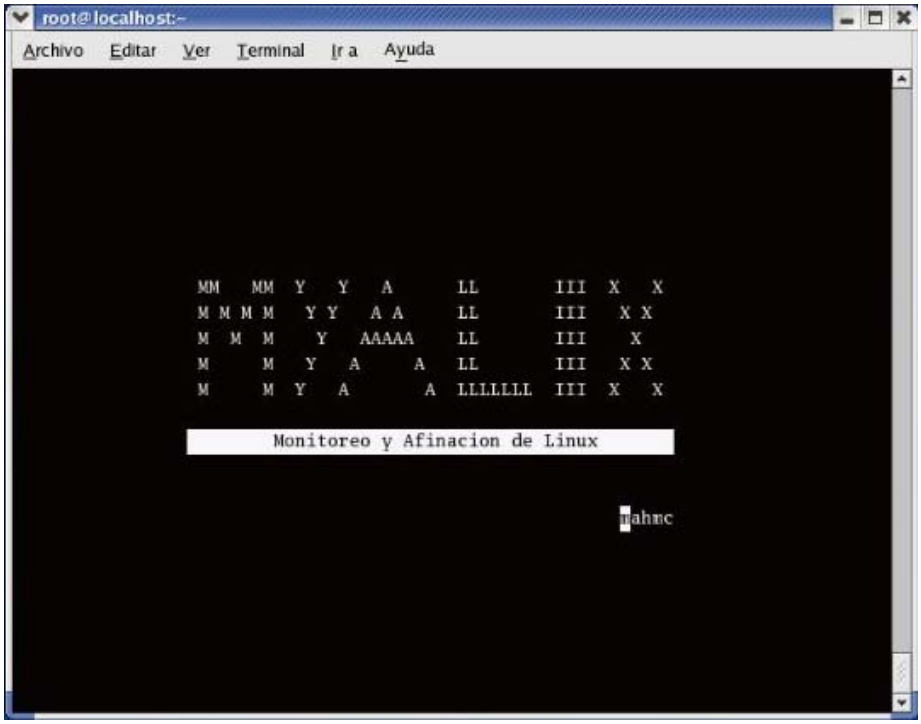
```
#  if [ -z "$OpcMain" ]
#  then
#     OpcMain="c"
#  fi

  case $OpcMain in
  [Cc]) tput cup 24 0
        LS="                    "
        RS="                    "
        echo -e "${B}${LS}Monitoreo del uso del Procesador ...${RS}${N}\c"
        sleep $SleepTime
        MyACpu
        ;;
  [Mm]) tput cup 24 0
        LS="                     "
        RS="                     "
        echo -e "${B}${LS}Monitoreo del uso de Memoria ...${RS}${N}\c"
        sleep $SleepTime
        MyAMem
        ;;
  [Dd]) tput cup 24 0
        LS="                     "
        RS="                      "
        echo -e "${B}${LS}Monitoreo del uso del Disco ...${RS}${N}\c"
        sleep $SleepTime
        MyADis
        ;;
  [Bb]) tput cup 24 0
        LS="                         "
        RS="                          "
        echo -e "${B}${LS}Monitoreo de Buffers ...${RS}${N}\c"
        sleep $SleepTime
        MyABuf
        ;;
  [Kk]) tput cup 24 0
        LS="                    "
        RS="                     "
        echo -e "${B}${LS}Monitoreo de Kolas de procesos ...${RS}${N}\c"
        sleep $SleepTime
        MyAKol
        ;;
  [Pp]) tput cup 24 0
        LS="                    "
        RS="                    "
        echo -e "${B}${LS}Monitoreo de Procesos existentes ...${RS}${N}\c"
        sleep $SleepTime
        MyAPro
        ;;
  [Ss]) ContinuaMain=false
        ;;
   *)   tput cup 24 0
        LS="                         "
        RS="                          "
```

```
        echo -e "${B}${LS}NO existe esa opcion${RS}${N}\a\c"
        sleep $SleepTime
        ;;
  esac
done
clear
echo -e "\n\nFin de myalix."
##################################################mahmc###
```

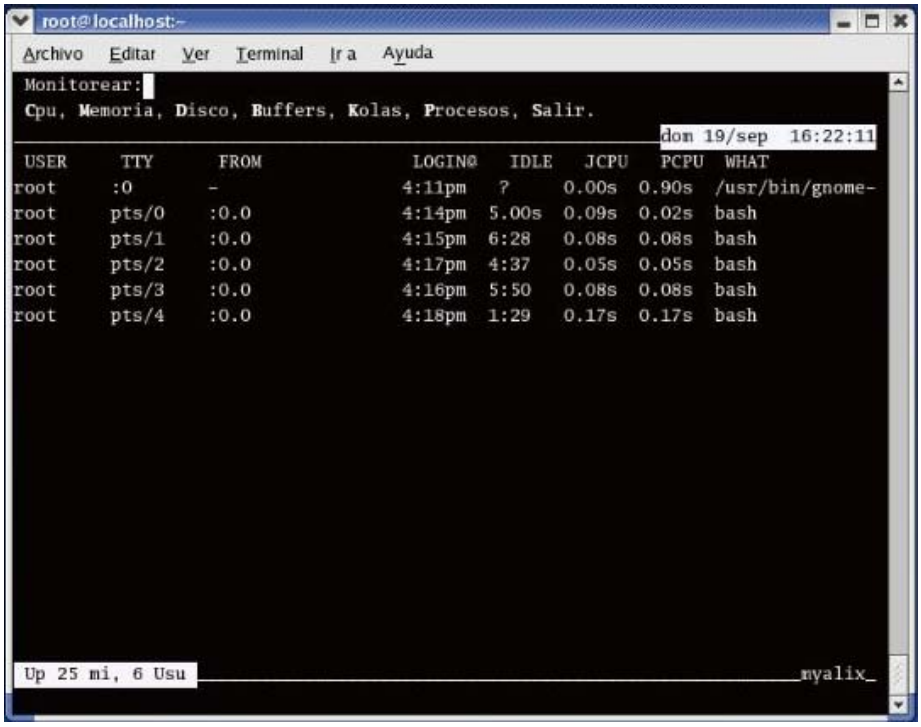
El *MenuIni* invocado por myalix, que está escrito en lenguaje C y que tiene por objeto delimitar las secciones de la pantalla y activar las opciones válidas del menú, se muestra a continuación:

```
/*************************************************************************
 * Programa: MenuIni.c
 * Objetivo: Desplegar el Menu de inicio de myalix
 * Lenguaje: C
 *
 * Autor   : Miguel Angel Herrera MC
 * Fecha de Creacion: Mayo del 2004                    Ultima modificacion: Ago del 2004
 *************************************************************************mahmc***/

# include <stdio.h>
# include <stdlib.h>
# include <string.h>

# define ESC '  '
# define LOWER 1
# define UPPER 25
# define STEP 1

main()

{
int reng,
      c;

system ("clear");
printf("  [1;1H Monitorear:");
printf("  [2;1H  [1mC  [0mpu,  [1mM  [0memoria,  [1mD  [0misco,  [1mB  [0muffers,
   [1mK  [0molas,  [1mP  [0mrocesos,  [1mS  [0malir.");
printf("  [3;1H_____
_____");
printf("  [24;1H_____
_____myalix_");

}/*end main*/
```

Al ejecutar el sistema con el comando myalix, se muestra la siguiente pantalla de presentación:



En seguida muestra los usuarios en sesión, la terminal que utilizan, el tiempo que tienen conectados al sistema, el comando que están ejecutando en ese momento y pregunta por el recurso que se desea monitorear.



Si revisamos el programa myalix, mostrado en éste capitulo, podemos percatarnos que dependiendo de la opción tecleada, se invocan a diferentes programas *shell* que están encargados de monitorear y recomendar acciones con el fin de afinar de la mejor manera posible los recursos del sistema operativo.

Los módulos de programación a los que invoca myalix son:

1. – MyACpu
   Encargado de monitorear y recomendar acciones para afinar el uso del Procesador Central.

2. – MyAMem
   Encargado de monitorear y recomendar acciones para afinar el uso de la Memoria.

3. – MyADis
   Encargado de monitorear y recomendar acciones para afinar el uso del Disco.

4. – MyABuf
   Encargado de monitorear y recomendar acciones para afinar el uso de los Buffers.

5. – MyAKol
   Encargado de monitorear y recomendar acciones para afinar el uso de la cola de procesos existentes.

6. – MyAProc
   Encargado de monitorear y recomendar acciones para afinar el uso de los procesos existentes.

1- MyACpu

Es el módulo encargado de monitorear y recomendar acciones para afinar del uso del procesador central del equipo que está siendo monitoreado, si los umbrales definidos para su desempeño han sido alcanzados o superados.

Es invocado al teclear la letra C (letra mayúscula de Cpu), desde el menú principal del sistema al preguntar por el factor que se desea monitorear.

Para poder realizarlo, verificamos el uso del CPU consultando el reporte de actividad del sistema, mismo que se actualiza cada diez minutos. Los comandos de administración utilizados se ejemplifican en el anexo del presente documento.

La información filtrada que nos interesa es la siguiente:

- El porcentaje de uso a nivel de usuario, es decir, porcentaje dedicado a las aplicaciones realizadas por los usuarios (%user).
- El porcentaje de uso ocurrido a nivel del usuario con cierta prioridad asignada (%nice).
- El porcentaje de uso a nivel del sistema (kernel), es decir, el porcentaje dedicado a atender peticiones propias del sistema (%system).
- El porcentaje de tiempo ocioso que presenta el procesador (%idle).

Umbrales de desempeño

Para conocer el desempeño del procesador central, se hace uso de los comandos de administración tales como sar, iostat, w y uptime, que se detallan en el anexo, y de los cuáles obtenemos los valores %user, %nice, %system y %idle.

Estos valores son comparados con los umbrales definidos por Hewlett Packard para maximizar su rendimiento, mismos que fueron recopilados a lo largo de varios años en el mantenimiento y afinación de sus equipos con sistema operativo hp-ux, que como sabemos, es uno de los más utilizados de los diferentes sabores existentes de Unix. Los umbrales a considerar aparecen en la parte inferior de la pantalla, en la sección de mensajes.

Si el tiempo ocioso (idle) del procesador central es muy bajo (debajo del 3%), quiere decir que la mayor parte del tiempo está dedicado a atender peticiones de los diferentes usuarios, así como del propio sistema operativo, presentándose de esta forma un cuello de botella en el uso del CPU, por lo que se deberá revisarse también la cantidad de llamadas realizadas al sistema.

Otro cuello de botella del CPU puede presentarse si tanto el tiempo dedicado a atender peticiones de los usuarios o al mismo sistema operativo supera el 80 por ciento, denotado por las variables %user y %system.

En el caso de que el CPU dedique mucho tiempo a atender las peticiones de los diferentes usuarios del equipo, se deberán revisar las aplicaciones que se están ejecutando para encontrar el por que están consumiendo tanto tiempo del procesador central, lo cuál generalmente se debe a una mala programación, mal uso de grandes localidades de memoria o a la falta de liberación de recursos ya utilizados.

Si la mayor parte del tiempo del procesador está dedicado al sistema, debemos revisar si es que se tiene excesivo intercambio (swap) de información o si el tamaño de buffers definido es el adecuado.

Opciones myalix

Una vez seleccionada la opción C (letra resaltada de Cpu) desde el menú principal de myalix, la primera línea del área de menús indicará que se está monitoreando el uso del CPU.

La segunda línea de ésta área de menús, mostrará las siguientes opciones que podrán ser seleccionadas:

User - Al teclear U la información desplegada será ordenada descendentemente por la columna que muestra el porcentaje de uso a nivel de usuario y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño anteriormente definidos.

Nice - Al teclear N la información desplegada será ordenada descendentemente por la columna que muestra el porcentaje de uso ocurrido a nivel del usuario con cierta prioridad asignada y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño anteriormente definidos en esta sección.

System - Al teclear S la información desplegada será ordenada descendentemente por la columna que muestra el porcentaje de uso a nivel del sistema y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño anteriormente definidos.

Idle **-** Al teclear I la información desplegada será ordenada descendentemente por la columna que muestra el porcentaje de tiempo ocioso y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño anteriormente definidos en ésta sección.

umBrales - Al teclear B podremos consultar y modificar los umbrales de funcionamiento definidos para el Cpu. Una vez modificados se actualizarán dichos umbrales en el área de mensajes al final de la pantalla.

Regresar - Al teclear R, regresaremos al menú principal de la aplicación, para poder monitorear otro elemento.

Si myalix detecta que alguno de los valores han alcanzado o superado los umbrales definidos para su correcto funcionamiento, inmediatamente enviará un mensaje indicando que se está presentando un cuello de botella en el CPU, mostrará una bitácora de eventos realzando aquellos que han rebasado los umbrales definidos y recomendará las acciones a realizar con el propósito de poder afinarlo.

Siempre que los umbrales sean superados, myalix creará una bitácora que será alojado en el directorio correspondiente para ser consultado posteriormente por el administrador del equipo.

En caso de presentarse un cuello de botella en el CPU debido a las aplicaciones desarrolladas por los usuarios, myalix solamente informará que se deberá revisar la forma de programación de dichas aplicaciones, ya que seguramente no están liberando adecuadamente recursos del sistema o están haciendo uso ineficiente de gran cantidad de memoria.

Programa MyACpu

El programa fuente realizado en lenguaje *shell* es un módulo separado, como todos los que forman el sistema, y se muestra a continuación:

```
#############################################################
# Programa: MyACpu
# Objetivo: Monitorear y Afinar el uso de la procesador del equipo.
#
#         Las opciones con las que cuenta son:
#         U - Ordena el uso del procesador por el porcentaje de usuario
#         N - Ordena el uso del procesador por el porcentaje Nice
#         S - Ordena el uso del procesador por el porcentaje del
#             sistema
#         I - Ordena el uso del procesador por el porcentaje idle
#         B - Muestra y permite cambiar los umbrales de desempeño
#         R - Para regresar al menu principal.
#
#        Cuando los umbrales de funcionamiento son alcanzados o
#        superados se escribe en la bitacora de eventos.
#
#        Los umbrales de funcionamiento para el CPU son:
#
#         % tiempo del usuario hasta el 80%  [ %User > 80 ]
```

```
#           % tiempo del systema hasta e 80%    [ %Sys  > 80 ]
#           % tiempo ocioso (idle) hasta el 3% [ %Idle <  3 ]
#
# Lenguaje: shell
# Autor   : Miguel Angel Herrera MC
# Fecha de Creacion: Mayo del 2004     Ultima modificacion: Ago. del 2004
##################################################mahmc###

#
# Constantes
#

B="`tput smso`"
N="`tput sgr0`"

#
# Variables
#

ContinuaCpu=true
UmbralIdle=3
UmbralUser=80
UmbralSystem=80
SleepTime=4
STime=3

#
# Funciones
#

EsperaEnter()
{
 tput cup 24 0
 LS="                    "
 RS="                    "
 echo -e "${B}${LS}Oprima <return> para continuar ...${RS}${N}\c"
 read -s ANSW
} #EsperaEnter

LimpiaDes()
{
 for REN in 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
 do
    tput cup $REN 0
    echo "                                                                    "
 done
} #end LimpiaDes

ExtraeUpCpu()
{
Contador=0
OutRange=`more UpCpu | wc -l`
#echo Lineas del Archivo UpCpu: $OutRange
```

```
#echo -e "\n\nMostrar el archivo por campos:"
Ren=10
tput cup $Ren 14 ; echo ${B}Hora${N}
tput cup $Ren 22 ; echo ${B}Cpu${N}
tput cup $Ren 32 ; echo ${B}%User${N}
tput cup $Ren 42 ; echo ${B}%Nice${N}
tput cup $Ren 52 ; echo ${B}%Sys${N}
tput cup $Ren 62 ; echo ${B}%Idle${N}
let Ren=Ren+1    ; tput cup $Ren 10
echo "${B} ------------------------------------------------------------ ${N}"

for i in `cat UpCpu | tail -7`
do
#   echo $i
   let Contador=Contador+1
   case $Contador in
      1) Hora=$i  ;;
      2) Cpu=$i   ;;
      3) PUser=$i ;;
      4) PNice=$i ;;
      5) PSys=$i  ;;
      6) PIdle=$i ;;
   esac
   if [ "$Contador" -eq 6 ] ; then
      let Ren=Ren+1
      tput cup $Ren 12 ; echo ${B}$Hora${N}
      tput cup $Ren 22 ; echo ${B}$Cpu${N}
      tput cup $Ren 32 ; echo ${B}$PUser${N}
      tput cup $Ren 42 ; echo ${B}$PNice${N}
      tput cup $Ren 52 ; echo ${B}$PSys${N}
      tput cup $Ren 62 ; echo ${B}$PIdle${N}
#      echo $Hora $Cpu $PUser $PNice $PSys $PIdle
#      read algo
      let Contador=0
   fi
done
let Ren=Ren+2
tput cup $Ren 12 ; echo ${B}Total: $OutRange${N}
} #end ExtraeUpCpu

Espacios()
{
# Parametros de entrada
Linea=""
Espacio=" "
NumEsp=$1

let CuentaEspacios=0
while [ "$CuentaEspacios" -le "$NumEsp" ]
do
   Linea=$Linea$Espacio
   let CuentaEspacios=CuentaEspacios+1
done
```

```
} #end Espacios

Marco()
{
# Parametros de entrada
let Ren1=$1 ; let Con1=$Ren1
let Col1=$2 ; let Con2=$Col1
let Ren2=$3+$Ren1
let Col2=$4
let TipoCuadro=$5

# Constantes
B="`tput smso`"
N="`tput sgr0`"

# Variables
Cadena=" "
Long=0

# Main Function
let Long=$Col2-$Col1
#tput cup 2 2 ; echo $Ren1, $Col1, $Ren2, $Col2, $Con1, $Con2, $Long

while [ "$Con1" -le "$Ren2" ]
do
#   while [ "$Con2" -le "$Col2" ]
#   do
#      echo $Con1 $Con2
#      tput cup $Con1 $Con2 ; echo "${B} ${N}"
      Cadena="                                                  "
      Cadena2="                                                 "
      Cadena3="+---------------------------------------------------------+"

#      Espacios $Long
#      Cadena=$Linea
#      tput cup $Con1 $Col1 ; echo "${B}$Cadena${N}"

      tput cup $Con1 $Col1
      case $TipoCuadro in
          1) echo "$Cadena"
          ;;
       2) if [ $Con1 = $Ren1 -o $Con1 = $Ren2  ] ; then
                echo "${B}$Cadena${N}"
          else
                echo "${B} ${N}$Cadena2${B} ${N}"
          fi
          ;;
          3) echo "${B}$Cadena${N}"
          ;;
       4) if [ $Con1 = $Ren1 -o $Con1 = $Ren2  ] ; then
                echo "${B}$Cadena3${N}"
          else
                echo "${B}|$Cadena2|${N}"
```

```
          fi
          ;;
       esac

#      let Con2=Con2+1
#   done
    let Con1=Con1+1
    let Con2=$Col1
done
} #end Marco

MensajeUpUmbral()
{
  case $1 in
   1) Mensaje="Debido al tiempo dedicado a procesos de los usuarios ..." ;;
   2) Mensaje="Debido al tiempo dedicado a procesos del sistema ..." ;;
   3) Mensaje="Debido al tiempo en ha estado oscioso ..." ;;
  esac

  tput cup 10 17
  echo -n "${B}                                        ${N}"
  tput cup 11 17
  echo -n "${B} El procesador ha rebasado umbrales definidos. ${N}"
# echo -n "${B} Se esta presentando un cuello de botella en el CPU ... ${N}"
# tput cup 12 17
# echo -e "${B} $Mensaje${N}\c"
  tput cup 12 17
  echo -n "${B}                                        ${N}"
  tput cup 24 0
  LS="                    "
  RS="                    "
  echo -e "${B}${LS}Espere un momento por favor ... ${RS}${N}\c"
  sleep $STime
} #end MensajeUpUmbral

BitacoraEventos()
{
 OutRange=`more UpCpu | wc -l`
 if [ "$OutRange" -gt 7 ] ; then
      let OutRange=7
 fi
 let LinMarco=OutRange+7
 Marco 7 10 $LinMarco 70 3
#Marco 7 10 16 70
 tput cup 8 29  ;  echo "${B} BITACORA DE EVENTOS ${N}"
 ExtraeUpCpu
 EsperaEnter
 Marco 8 11 $LinMarco 77 4
 case $1 in
#    Acciones a realizar al superarse los umbrales definidos para el CPU.
 1) tput cup 8 21 ;  echo "${B} ACCIONES A REALIZAR ${N}"
#    tput cup 11 15 : echo "${B}----------------------${N}"
    tput cup 10 19 ; echo "${B}1.- Revisar las  aplicaciones  desarrolladas por ${N}"
```

```
    tput cup 11 19 ; echo "${B}    los usuarios, ya que estan consumiendo mucho${N}"
    tput cup 12 19 ; echo "${B}    tiempo del procesador.                    ${N}"
    EsperaEnter ;;
 2) tput cup 8 21 ;  echo "${B} ACCIONES A REALIZAR ${N}"
    tput cup 10 19 ; echo "${B}1.- Revisar los procesos que se están ejecutando.${N}"
    tput cup 11 19 ; echo "${B}2.- Revisar el uso de la memoria.${N}"
    tput cup 12 19 ; echo "${B}3.- Revisar el uso del disco.${N}"
    tput cup 13 19 ; echo "${B}4.- Revisar la cola de procesos.${N}"
    EsperaEnter ;;
 3) tput cup 8 21 ;  echo "${B} ACCIONES A REALIZAR ${N}"
    tput cup 10 19 ; echo "${B}1.- Revisar los procesos que se están ejecutando.${N}"
    tput cup 11 19 ; echo "${B}2.- Revisar la cola de procesos con prioridad alta.${N}"
    tput cup 12 19 ; echo "${B}3.- Verificar la prioridad de los procesos back ground..${N}"
    EsperaEnter ;;
 esac
} #end BitacoraEventos

#
# Programa Principal
#

while $ContinuaCpu
do
  MenuCpu
  Usuarios=`who | wc -l `
# tput cup 1 60
# echo "${Usuarios} Usuarios"
  let Usuarios=Usuarios+10
  let Usuarios=Usuarios-10
  tput cup 2 60
  echo "${B}`date +"%a %d/%b  %H:%M:%S"`${N}"
  tput cup 23 0
# echo "${B} Up`uptime | cut -c14-19  `${N}"
  echo "${B} Up`uptime|cut -c14-19`, ${Usuarios} Usu ${N}"
  tput cup 23 30 ; echo "${B}%User>${UmbralUser} %Sys>${UmbralSystem}
%Idle<${UmbralIdle}${N}"
  tput cup 3 3  ; echo Hora
  tput cup 3 17 ; echo CPU
  tput cup 3 26 ; echo %User
  tput cup 3 34 ; echo %Nice
  tput cup 3 41 ; echo %System
  tput cup 3 50 ; echo %Idle

  sar -u > saru
  lineas=`more saru | wc -l`
  let lineas=lineas-3
  tail -$lineas saru > saru2
  let lineas=lineas-1
  head -$lineas saru2 > saru
  tput cup 4 0
  awk '{printf " %8s\t%4s\t%7.2f\t%7.2f\t%7.2f\t%7.2f\n",$1,$2,$3,$4,$5,$6 }' saru | sort -gr |
head -19
```

```
  tput cup 0 16
  read OpcCpu

  case $OpcCpu in
  [Uu]) LimpiaDes
        tput cup 3 3  ;  echo %User
        tput cup 3 13 ;  echo %Nice
        tput cup 3 22 ;  echo %System
        tput cup 3 33 ;  echo %Idle
        tput cup 3 44 ;  echo Hora
        awk '{printf " %7.2f   %7.2f   %7.2f   %7.2f    %8s\n",$3,$4,$5,$6,$1 }' saru | sort -gr |
head -19
        awk -v lim=$UmbralUser '$3 > lim   { print $0 > "UpCpu" }'  saru
        sleep $SleepTime
        if [ -e UpCpu ]
        then
          MensajeUpUmbral 1
          BitacoraEventos 1
          EscBit 1 UpCpu
          rm UpCpu
        fi
        ;;
  [Nn]) LimpiaDes
        tput cup 3 3  ;  echo %Nice
        tput cup 3 13 ;  echo %User
        tput cup 3 22 ;  echo %System
        tput cup 3 33 ;  echo %Idle
        tput cup 3 44 ;  echo Hora
        awk '{printf " %7.2f   %7.2f   %7.2f   %7.2f    %8s\n",$4,$3,$5,$6,$1 }' saru | sort -gr |
head -19
        sleep $SleepTime
        ;;
  [Ss]) LimpiaDes
        tput cup 3 2  ;  echo %System
        tput cup 3 13 ;  echo %Nice
        tput cup 3 23 ;  echo %User
        tput cup 3 33 ;  echo %Idle
        tput cup 3 44 ;  echo Hora
        awk '{printf " %7.2f   %7.2f   %7.2f   %7.2f    %8s\n",$5,$4,$3,$6,$1 }' saru | sort -gr |
head -19
        awk -v lim=$UmbralSystem  '$5 > lim   { print $0 > "UpCpu" }'  saru
        sleep $SleepTime
        if [ -e UpCpu ]
        then
          MensajeUpUmbral 2
          BitacoraEventos 2
          EscBit 2 UpCpu
          rm UpCpu
        fi
        ;;
  [Ii]) LimpiaDes
        tput cup 3 3  ;  echo %Idle
        tput cup 3 13 ;  echo %Nice
```

```
        tput cup 3 22 ;  echo %System
        tput cup 3 33 ;  echo %User
        tput cup 3 44 ;  echo Hora
        awk '{printf " %7.2f   %7.2f   %7.2f   %7.2f    %8s\n",$6,$4,$5,$3,$1 }' saru | sort -gr |
head -19
        awk -v lim=$UmbralIdle ' $6 < lim   { print $0 > "UpCpu" }'  saru
        sleep $SleepTime
        if [ -e UpCpu ]
        then
          MensajeUpUmbral 3
          BitacoraEventos 3
          EscBit 3 UpCpu
          rm UpCpu
        fi
        ;;
  [Bb]) LimpiaDes
        Marco 4 8 10 63 3
        tput cup 5 9 ; echo ${B}UMBRALES DEL DESEMPE├O DEL PROCESADOR${N}
           tput cup 7 13 ; echo "${B}Tiempo Usuario > ${N}" $UmbralUser %
           tput cup 8 13 ; echo "${B}Tiempo Sistema > ${N}" $UmbralSystem %
           tput cup 9 13 ; echo "${B}Tiempo Ocioso  < ${N} " $UmbralIdle  %
        tput cup 7 31 ; read UmbralCpu
        if [ -n "$UmbralCpu" ]
        then
           UmbralUser=$UmbralCpu
        fi
        tput cup 8 31 ; read UmbralCpu
        if [ -n "$UmbralCpu" ]
        then
           UmbralSystem=$UmbralCpu
        fi
        tput cup 9 32 ; read UmbralCpu
        if [ -n "$UmbralCpu" ]
        then
           UmbralIdle=$UmbralCpu
        fi
        ;;
  [Rr]) ContinuaCpu=false
        ;;
   *)   tput cup 24 0
        LS="                              "
        RS="                              "
        echo -e "${B}${LS}NO existe esa opcion${RS}${N}\a\c"
        sleep $SleepTime
        ;;
  esac
done
rm saru
rm saru2
return
#####################################################mahmc###
```

El *MenuCpu* invocado por MyACpu, que está escrito en lenguaje C y que tiene por objeto respetar las secciones creadas en la pantalla y activar las opciones válidas del menú para monitorear al procesador central, se muestra a continuación:

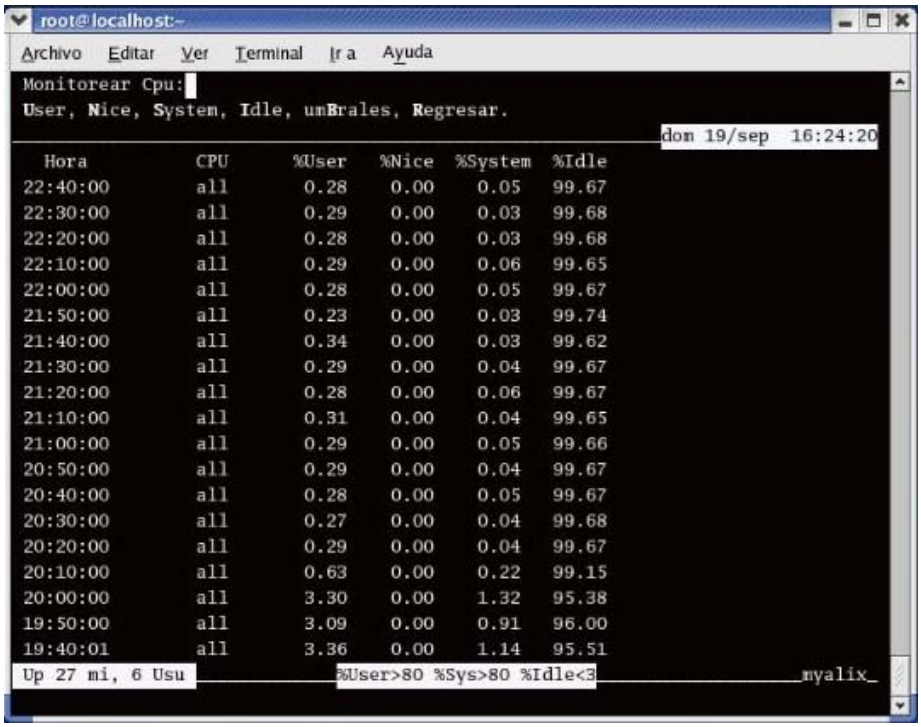
```
/*************************************************************************
 * Programa: MenuCpu
 * Objetivo: Desplegar el Menu de monitoreo del CPU
 * Lenguaje: C
 *
 * Autor   : Miguel Angel Herrera MC
 * Fecha de Creacion: Mayo del 2004                    Ultima modificacion: Ago del 2004
 ************************************************************************mahmc***/

# include <stdio.h>
# include <stdlib.h>
# include <string.h>

# define ESC '  '
# define LOWER 1
# define UPPER 25
# define STEP 1

main()

{
int reng,
      c;

system ("clear");
printf("  [1;1H Monitorear Cpu:");
printf("  [2;1H  [1mU  [0mser,  [1mN  [0mice,  [1mS  [0mystem,  [1mI  [0mdle,
um  [1mB  [0mrales,  [1mR  [0megresar.");
printf("  [3;1H_____________________________________________________________
_____________");
printf("  [24;1H_____________________________________________________________
______myalix_");

}/*end main*/
```

Los demás módulos que componen el sistema y que son: *MyAMem, MyADis, MyABuf, MyAKol* y *MyAProc* tienen una estructura similar al *MyACpu*, incluso invocan a sus respectivos menús escritos también en lenguaje C, siempre respetando las secciones definidas en la pantalla.

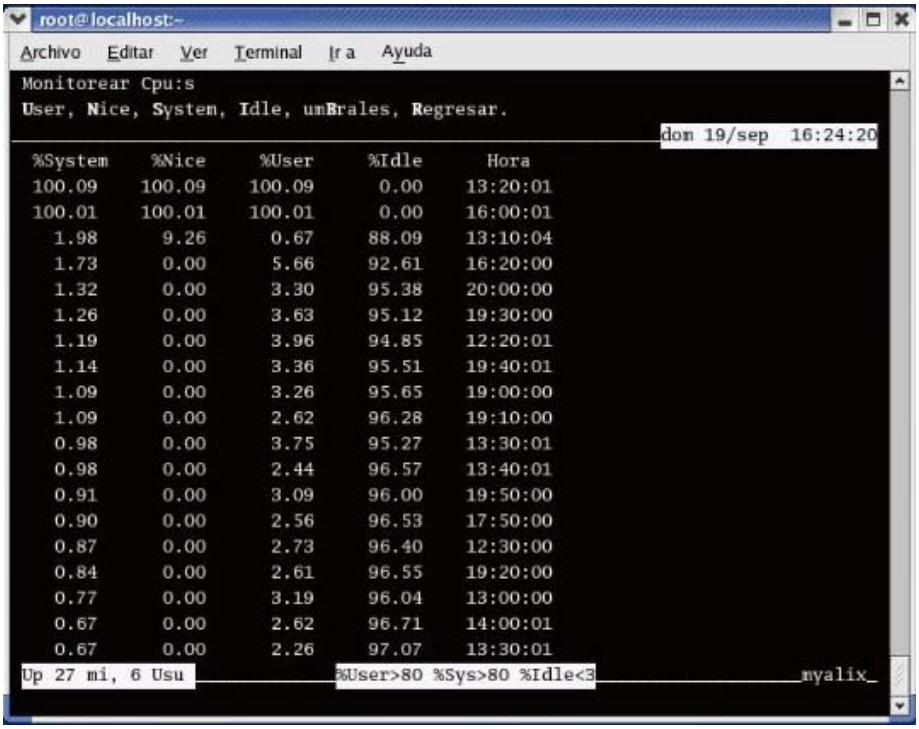
Por tal motivo no consideramos menester incluir el código fuente de todos y cada uno de los módulos del sistema, ya que entregamos con el presente trabajo un disco compacto con todos los programas realizados, mismos que en conjunto nos dan un gran total de más de tres mil líneas de código tanto en programación *shell* (la mayoría) como en lenguaje C (los menús de opciones de cada módulo).

Ejecución del Módulo

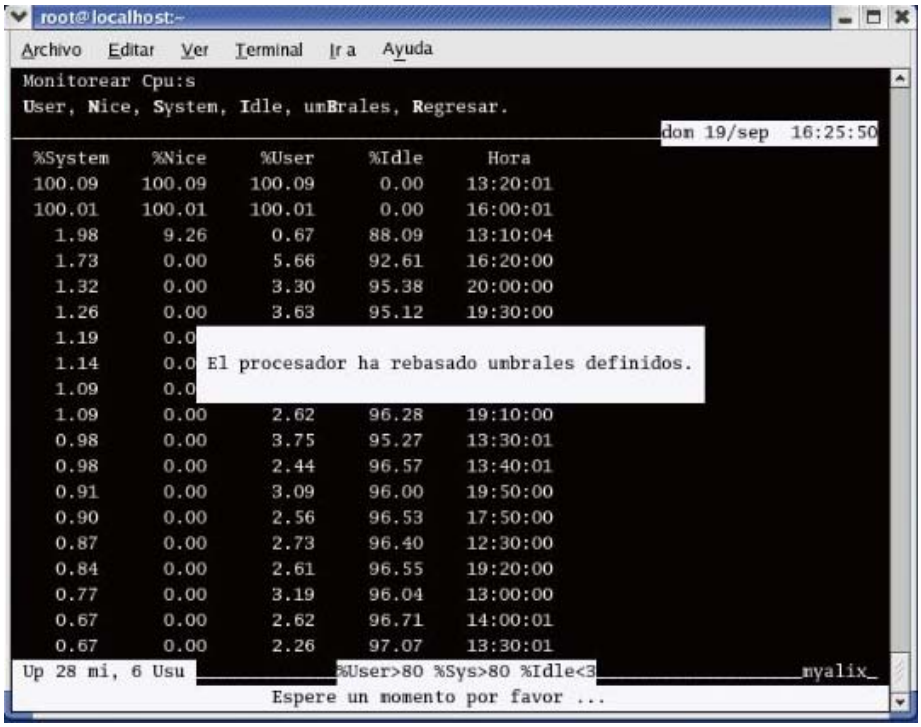
Se invoca con la letra C (Cpu) desde el menú principal y presenta la siguiente pantalla:



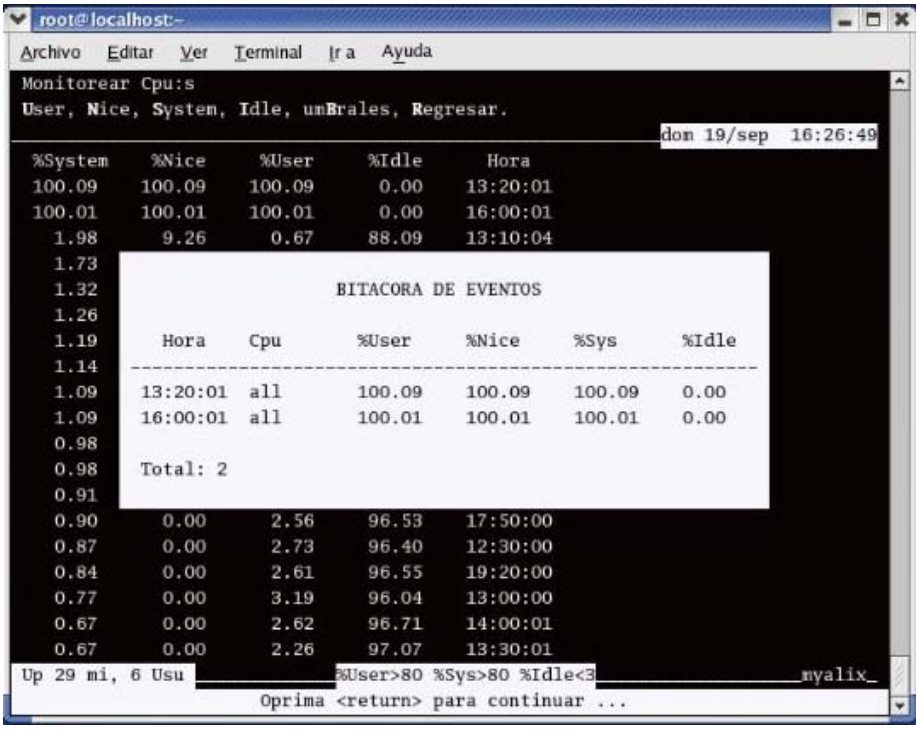
Al seleccionar la letra S (System), ordena descendentemente la información mostrada por el porcentaje de uso del sistema, de la siguiente forma:



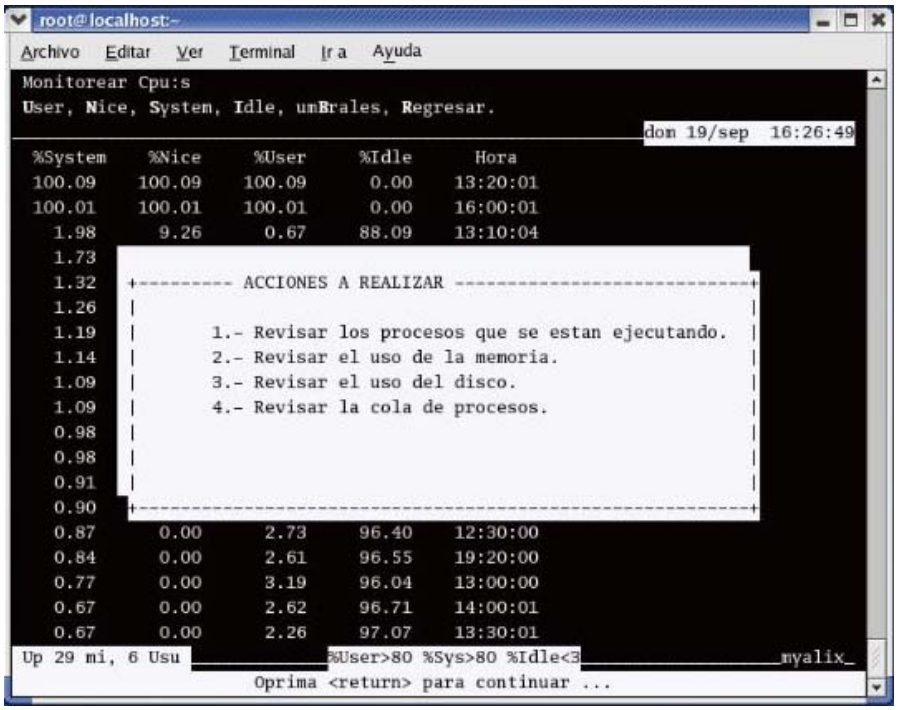
Los valores del porcentaje de uso son comparados con los umbrales definidos (mostrados en la parte inferior de la pantalla) y si alguno de ellos rebasa dicho umbral, se recibirá el siguiente mensaje:



Inmediatamente después mostrará la bitácora de eventos con cada uno de los valores que superaron dichos umbrales.



Al oprimir *return* (tal y como se pide en la sección de mensajes), se definirán las acciones a realizar para afinar el uso del procesador.



Cada vez que se presente la bitácora de eventos, será almacenada en un archivo dentro del directorio bitácoras con el nombre formado por el día, el mes y hora del suceso, para su posterior análisis.

Todos los módulos del sistema myalix, al presentar la información del recurso a monitorear, comparan los valores mostrados con los umbrales definidos para cada recurso, localizados en la parte inferior de la pantalla.

Cuando dichos valores rebasan los umbrales definidos, el sistema realiza las acciones aquí ejemplificadas, mismas que ya no serán mostradas en cada uno de los siguientes módulos a detallar, evitando de ésta forma llenar el presente capítulo solamente de pantallas de mensajes.

2. - MyAMem

Es el módulo encargado de monitorear y recomendar acciones para afinar en la manera de lo posible, el uso de la memoria principal en el equipo que está siendo monitoreado, si los umbrales definidos para su desempeño han sido alcanzados o superados.

Es invocado al teclear la letra M (que es la letra mayúscula de Memoria), desde el menú principal del sistema al preguntar por el recurso que se desea monitorear.

Para poder realizarlo, verificamos el uso de la memoria principal ocupando rutinas de administración propias de Linux, ejemplificadas en el anexo del presente documento y las cuales ofrecen la siguiente información:

- Cantidad de memoria disponible, expresada en Kilobytes (%kbmemfree)
- Cantidad de memoria utilizada, expresada en Kilobytes (%kbmemused)
- Porcentaje de uso de la memoria (%memused)
- Cantidad de memoria utilizada en buffers por el sistema operativo, expresada en Kilobytes (%kbbuffers)
- Cantidad de espacio disponible del área de swap, expresada en Kilobytes (%kbswpfree)
- Cantidad de espacio utilizado en del área de swap, expresada en Kilobytes (%kbswpused)
- Porcentaje de uso del área de swap, expresado en Kilobytes (%swpused)

Umbrales de desempeño.

Para conocer el desempeño de la memoria principal, se hace uso de los comandos de administración tales como sar, vmstat y free, que se detallan en el anexo, y de los cuáles se obtienen los valores %kbmemfree, %kbmemused, %memused, %swap entre otros.

Estos valores son comparados con los umbrales definidos por Hewlett Packard para maximizar su rendimiento, mismos que fueron recopilados a lo largo de varios años en el mantenimiento y afinación de sus equipos con sistema operativo hp-ux, que como sabemos, es uno de los más utilizados de los diferentes sabores existentes de Unix.

Si el porcentaje de memoria principal utilizada (%memused) o el porcentaje de uso del área swap superan el 80 por ciento, el sistema podría pasmarse por falta de memoria de trabajo, presentándose de esta forma un cuello de botella en el uso de la memoria principal del equipo.

Recordemos que es recomendable disponer hasta un 80 por ciento del total de la memoria para uso de aplicaciones en el equipo. Solamente en caso de realizar *benchmarks* sería recomendable incrementar este porcentaje de uso, considerando los potenciales problemas que podrían presentarse.

Al mencionar el uso de la memoria, debemos considerar dos conceptos relacionados con la misma, como son la paginación y swapeo (intercambio). La memoria está manejada por páginas, las cuáles generalmente tiene un tamaño de 2 kb dependiendo de la arquitectura del equipo.

Cuando el proceso es atendido, se cargan en la memoria tanto datos como estructuras cuando son requeridas, siempre por páginas. Así la paginación será la carga o descarga de datos o estructuras a o de la memoria principal.

En la mayoría de los sistemas multiusuarios se maneja el concepto de memoria virtual, que es la conjunción de la memoria física del sistema y un área de disco dedicada al uso de memoria, a la cuál se le ha denominado swap y la cuál permite el intercambio de páginas de memoria entre la memoria principal y el disco. A este intercambio de información se le conoce como *swapeo*.

Otro cuello de botella que puede presentarse en el manejo de la memoria se conoce como *thrashing* y consiste en que el procesador central gasta más tiempo en realizar paginaciones que en realmente procesar la información.

Opciones myalix

Una vez seleccionada la opción M (letra mayúscula de Memoria) desde el menú principal de myalix, la primera línea del área de menús indicará que se está monitoreando el uso de la memoria. En la sección de mensajes aparecerán los umbrales óptimos de funcionamiento del uso de la memoria.

La segunda línea de ésta área de menús, mostrará las siguientes opciones que podrán ser seleccionadas:

Disponible - Al teclear D la información desplegada será ordenada descendentemente por la columna que muestra la cantidad de memoria disponible y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos.

Usada - Al teclear U la información desplegada será ordenada descendentemente por la columna que muestra la cantidad de memoria utilizada y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos en esta sección.

Porcentaje - Al teclear P la información desplegada será ordenada descendentemente por la columna que muestra el porcentaje de memoria utilizada y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos.

Swap - Al teclear S la información desplegada será ordenada descendentemente por la columna que muestra el porcentaje de área swap utilizada y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos.

umBrales - Al teclear B podremos consultar y modificar los umbrales de funcionamiento definidos para la memoria. Una vez modificados se actualizarán dichos umbrales en el área de mensajes al final de la pantalla.

Regresar - Al teclear R, regresaremos al menú principal de la aplicación, para poder monitorear otro elemento.

Si myalix detecta que alguno de los valores han alcanzado o superado los umbrales definidos para su correcto funcionamiento, inmediatamente enviará un mensaje indicando que se está presentando un cuello de botella en la memoria, mostrará una bitácora de eventos realzando aquellos que han rebasado los umbrales definidos y recomendará las acciones a realizar con el propósito de poder afinarlo.

Siempre que los umbrales sean superados, myalix creará una bitácora que será alojado en el directorio correspondiente para ser consultado posteriormente por el administrador del equipo.

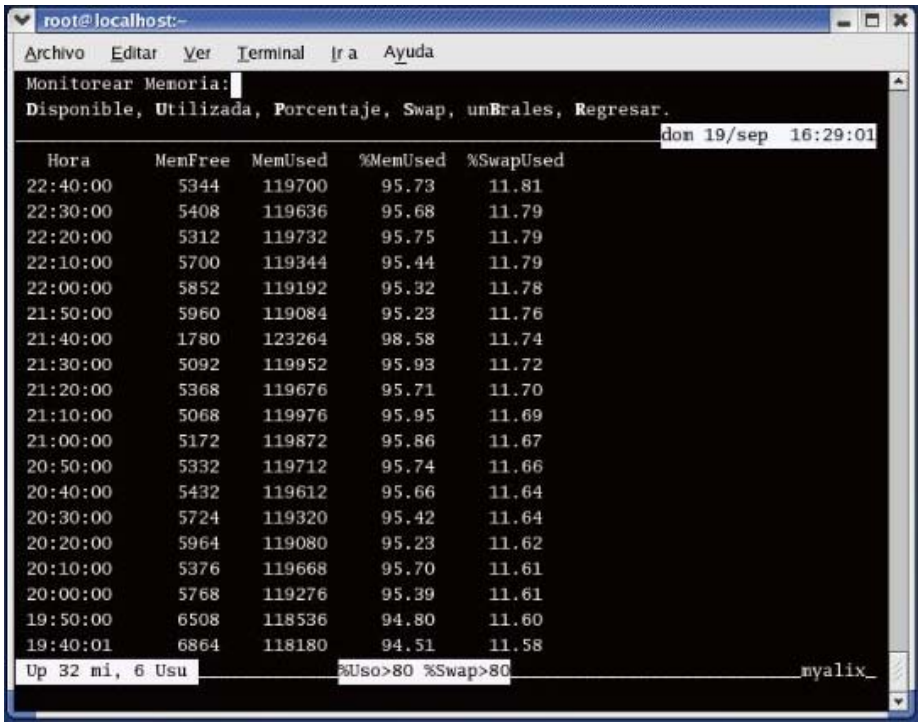
Programa MyAMem

Está realizado en programación *shell*, invoca al menú de opciones *MenuMem* escrito en lenguaje C y se encuentra localizado en el directorio monitor, dentro del disco compacto entregado con el presente trabajo.

Tiene una estructura similar del programa *MyACpu*, mostrado en éste capítulo.

Ejecución del Módulo

Se invoca con la letra M (Memoria) desde el menú principal y presenta la siguiente pantalla:



Cuando los valores mostrados rebasan los umbrales definidos en la parte inferior de la pantalla, el sistema informa al usuario del suceso, crea la bitácora de eventos e indica las acciones a realizar con el fin de poder afinar de la mejor forma posible la memoria de trabajo, tal y como lo ejemplificamos en el módulo MyACpu.

3. - MyADis

Es el módulo encargado de monitorear y recomendar acciones para afinar el uso del disco existente en el equipo que está siendo monitoreado, si los umbrales definidos para su desempeño han sido alcanzados o superados.

Es invocado al teclear la letra D (letra mayúscula de Disco), desde el menú principal del sistema al preguntar por el recurso que se desea monitorear.

Para poder realizarlo, verificamos el uso del disco ocupando rutinas de administración propias de Linux, ejemplificadas en el anexo del presente documento y las cuales ofrecen la siguiente información:

- Número de transacciones emitidas por segundo hacia el dispositivo (%tps)
- Número de sectores transferidos de o hacia el dispositivo, considerando que los sectores tienen un tamaño de 512 Kilobytes (%sect/s)
- Porcentaje de espacio disponible en el disco (%diskfree)

Umbrales de desempeño.

Para conocer el desempeño del o de los discos disponibles, se hace uso de los comandos de administración tales como df, iostat y sar, que se detallan en el anexo, y de los cuáles se obtienen los valores %tps, %sect/s y %diskfree entre otros.

Estos valores son comparados con los umbrales definidos por Hewlett Packard para maximizar su rendimiento, mismos que fueron recopilados a lo largo de varios años en el mantenimiento y afinación de sus equipos con sistema operativo hp-ux, que como sabemos, es uno de los más utilizados de los diferentes sabores existentes de Unix.

Si el porcentaje de uso de disco utilizado (%diskused) supera el 93 por ciento, el sistema podría pasmarse por falta de memoria de almacenamiento, presentándose de esta forma un cuello de botella en el uso del disco. En tal caso se deberá depurar el disco de almacenamiento y liberar tanto espacio como sea posible. No olvidemos que un espacio del disco está dedicado al área de swap.

Otro cuello de botella que puede presentarse en el disco, es si el porcentaje de disco disponible es menor al siete por ciento. Los umbrales de funcionamiento serán mostrados en el área de mensajes en la parte inferior de la pantalla.

En el caso de contar con más de un disco se deberá procurar balancear las aplicaciones y los datos residentes entre los discos existentes.

Opciones myalix

Una vez seleccionada la opción D (letra mayúscula de Disco) desde el menú principal de myalix, la primera línea del área de menús indicará que se está monitoreando el uso del disco. Los umbrales de funcionamiento óptimo del disco serán mostrados al final de la pantalla en la sección de mensajes.

La segunda línea de ésta área de menús, mostrará las siguientes opciones que podrán ser seleccionadas:

Transacciones - Al teclear T la información desplegada será ordenada descendentemente por la columna que muestra el número de transacciones emitidas por segundo hacia el dispositivo y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos.

Sectores - Al teclear S la información desplegada será ordenada descendentemente por la columna que muestra el número de sectores transferidos y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos en esta sección.

Disponible - Al teclear D la información desplegada será ordenada descendentemente por el espacio aún disponible en el disco de almacenamiento utilizado.

Utilizado - Al teclear U la información desplegada será ordenada descendentemente por el espacio utilizado en el disco.

Porcentaje - Al teclear P la información desplegada será ordenada descendentemente por el porcentaje de espacio utilizado en el disco.

umBrales - Al teclear B podremos consultar y modificar los umbrales de funcionamiento definidos para el disco. Una vez modificados se actualizarán dichos umbrales en el área de mensajes al final de la pantalla.

Regresar - Al teclear R, regresaremos al menú principal de la aplicación, para poder monitorear otro elemento.

Si myalix detecta que alguno de los valores han alcanzado o superado los umbrales definidos para su correcto funcionamiento, inmediatamente enviará un mensaje indicando que se está presentando un cuello de botella en el disco, mostrará una bitácora de eventos realzando aquellos que han rebasado los umbrales definidos y recomendará las acciones a realizar con el propósito de poder afinarlo.

Siempre que los umbrales sean superados, myalix creará una bitácora que será alojado en el directorio correspondiente para ser consultado posteriormente por el administrador del equipo.

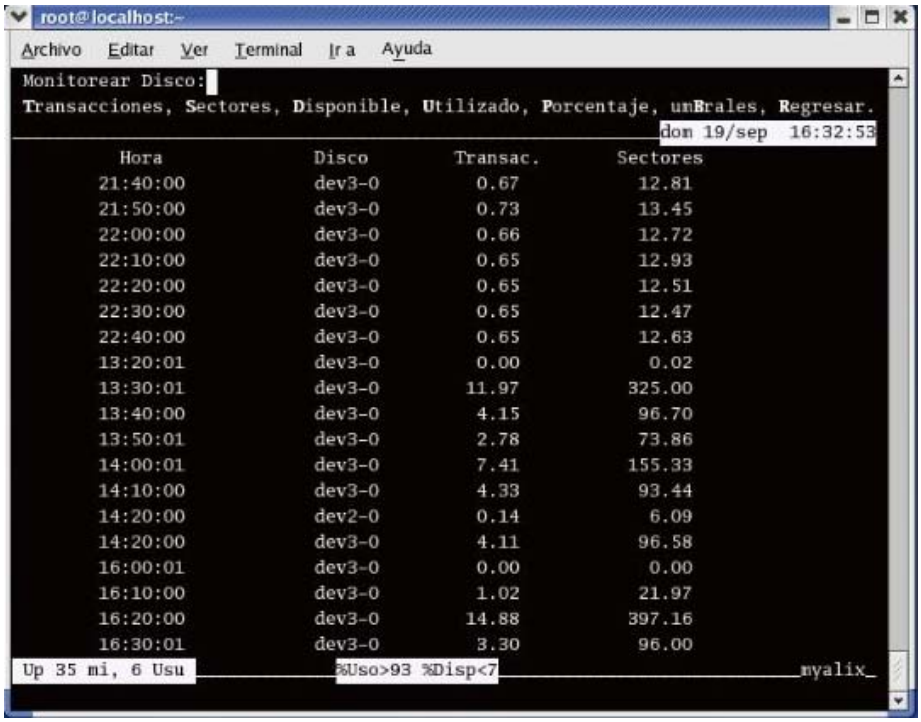
En caso de que se presente un cuello de botella en el disco y no se pueda remover ninguna aplicación, será entonces inevitable el considerar el incorporar un nuevo dispositivo de almacenamiento al equipo en cuestión o cambiar el actual por uno de mayor capacidad.

Programa MyADis

Está realizado en programación *shell*, invoca al menú de opciones *MenuDis* escrito en lenguaje C y se encuentra localizado en el directorio monitor, dentro del disco compacto entregado con el presente trabajo. Tiene una estructura similar del programa *MyACpu*, mostrado en éste capítulo.

Ejecución del Módulo

Se invoca con la letra D (Disco) desde el menú principal y presenta la siguiente pantalla:



Cuando los valores mostrados rebasan los umbrales definidos en la parte inferior de la pantalla, el sistema informa al usuario del suceso, crea la bitácora de eventos e indica las acciones a realizar con el fin de poder afinar de la mejor forma posible el uso del disco o memoria de almacenamiento, tal y como lo ejemplificamos en el módulo MyACpu.

4. - MyABuf

Es el módulo encargado de monitorear y recomendar acciones para afinar las operaciones de los buffers existentes en el equipo que está siendo monitoreado, si los umbrales definidos para su desempeño han sido alcanzados o superados.

Es invocado al teclear la letra B (letra mayúscula de Buffers), desde el menú principal del sistema al preguntar por el recurso que se desea monitorear.

Para poder realizarlo, verificamos el uso de los buffers consultando el reporte de actividad del sistema, mismo que se actualiza cada diez minutos. Los comandos de administración utilizados se ejemplifican en el anexo del presente documento.

La información filtrada que nos interesa es la siguiente:

- Número total de transferencias emitidas por segundo al disco físico (%tps)
- Número total de peticiones de lectura por segundo al disco físico (%rtps)
- Número total de escrituras por segundo emitidas al disco físico (%wtps)
- Número total de bloques leídos por segundo (%bread)
- Número total de bloques escritos por segundo (%bwrtn)

Umbrales de desempeño.

Para conocer el desempeño de los buffers disponibles, se hace uso de los comandos de administración tales como iostat y sar, que se detallan en el anexo del presente documento, y de los cuáles se obtienen los valores %tps, %rtps, %wtps, %bread y %bwrtn.

Estos valores son comparados con los umbrales definidos por Hewlett Packard para maximizar su rendimiento, mismos que fueron recopilados a lo largo de varios años en el mantenimiento y afinación de sus equipos con sistema operativo hp-ux, que como sabemos, es uno de los más utilizados de los diferentes sabores existentes de Unix.

Los valores que deben ser considerados, son los denominados *cache hit ratios (*tanto de lectura como de escritura) que son calculados de la siguiente forma:

$$\%rcache = (1 - bread / lread) * 100)$$
$$\%wcache = (1 - bwrit / lwrit) * 100)$$

Para tener un adecuado desempeño de los buffers, %rcache no debe estar por debajo del 90 por ciento y el %wcache no debe estar por debajo del 70 o 75 por ciento.

Si los porcentajes son menores a los estipulados, debemos revisar que clase de lecturas y escrituras está realizando el sistema, con lo cuál inevitablemente el tamaño de los buffers deberá ser ampliado, mejorando de esta forma su desempeño.

Opciones myalix

Una vez seleccionada la opción B (letra mayúscula de Buffers), desde el menú principal de myalix, la primera línea del área de menús indicará que se está monitoreando el uso de los buffers. Los umbrales de funcionamiento óptimo de los buffers serán mostrados al final de la pantalla en la sección de mensajes.

La segunda línea de ésta área de menús, mostrará las siguientes opciones que podrán ser seleccionadas:

Transacciones - Al teclear T la información desplegada será ordenada descendentemente por la columna que muestra el número total de transacciones por segundo y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos.

Lecturas - Al teclear L la información desplegada será ordenada descendentemente por la columna que muestra el número total de peticiones de lectura y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos en esta sección.

Escrituras - Al teclear E la información desplegada será ordenada descendentemente por la columna que muestra el número total de escrituras y realizará el análisis de cada uno de los valores mostrados, comparándolos con los umbrales de desempeño definidos en esta sección.

rCache **-** Al teclear C la información desplegada será ordenada descendentemente por el porcentaje de lecturas cache entre los buffers del sistema y los dispositivos.

Wcache **-** Al teclear W la información desplegada será ordenada descendentemente por el porcentaje de escrituras cache entre los buffers del sistema y los dispositivos de bloques existentes en el equipo.

umBrales - Al teclear B podremos consultar y modificar los umbrales de funcionamiento definidos para el disco. Una vez modificados se actualizarán dichos umbrales en el área de mensajes al final de la pantalla.

Regresar - Al teclear R, regresaremos al menú principal de la aplicación, para

poder monitorear otro elemento.

Si myalix detecta que alguno de los valores han alcanzado o superado los umbrales definidos para su correcto funcionamiento, inmediatamente enviará un mensaje indicando que se está presentando un cuello de botella en los buffers, mostrará una bitácora de eventos realzando aquellos que han rebasado los umbrales definidos y recomendará las acciones a realizar con el propósito de poder afinarlo.

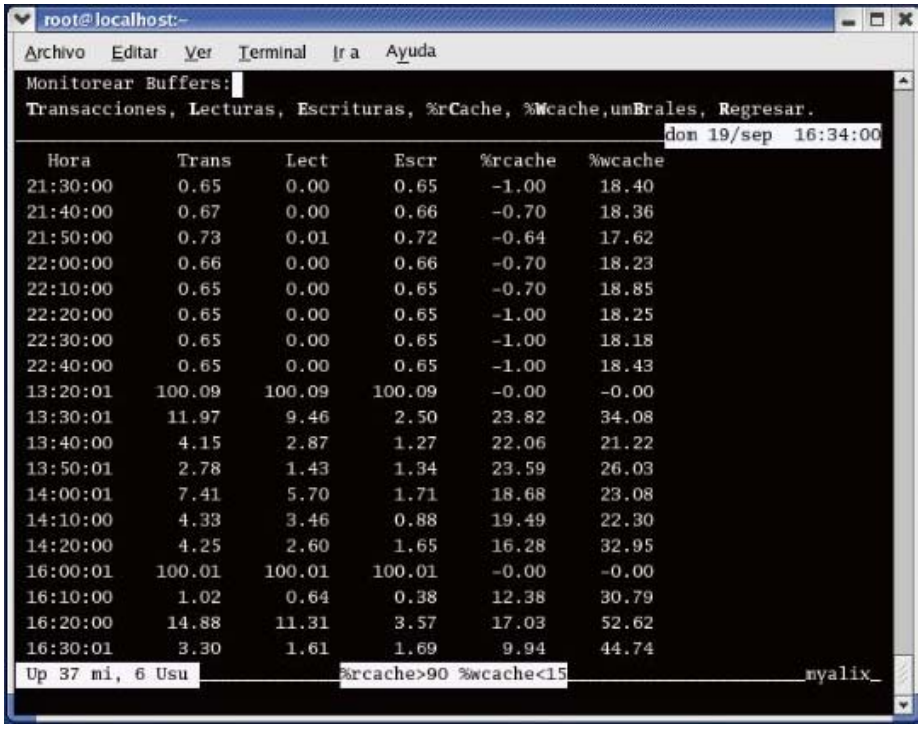
Siempre que los umbrales sean superados, myalix creará una bitácora que será alojado en el directorio correspondiente para ser consultado posteriormente por el administrador del equipo. En caso de que se presente un cuello de botella en el uso de buffers, seguramente el administrador deberá reiniciar el sistema en modo monousuario y tendrá que reconfigura el tamaño de los buffers antes de que los usuarios puedan hacer uso nuevamente del sistema.

Programa MyABuf

Está realizado en programación *shell*, invoca al menú de opciones *MenuBuf* escrito en lenguaje C y se encuentra localizado en el directorio monitor, dentro del disco compacto entregado con el presente trabajo. Tiene una estructura similar del programa *MyACpu*, mostrado en éste capítulo.

Ejecución del Módulo

Se invoca con la letra B (Buffers) desde el menú principal y presenta la siguiente pantalla:



5. - MyAKol

Es el módulo encargado de monitorear y recomendar acciones para afinar la cola de procesos existente en el equipo que está siendo monitoreado, si los umbrales definidos para su desempeño han sido alcanzados o superados.

Es invocado al teclear la letra K (letra mayúscula de Kolas), desde el menú principal del sistema al preguntar por el recurso que se desea monitorear.

Para poder realizarlo, se verifica la cola de procesos existentes en el equipo ocupando rutinas de administración propias de Linux, ejemplificadas en el anexo del presente documento y las cuales ofrecen la siguiente información:

- Número total de procesos que se encuentran en espera de ser ejecutados (enEspera).
- Número total procesos que se encuentran en lista por ser atendidos (enLista).
- El porcentaje de carga que se está presentando en el equipo, en el último minuto (Carga1)
- El porcente de carga que se está presentando en el equipo, en los últimos cinco minutos (Carga5)

Umbrales de desempeño.

Para conocer el estado de los procesos del sistema, hacemos uso de los comandos de administración tales como ps, free y sar, que se detallan en el anexo del presente documento, y de los cuáles obtenemos los valores enEspera, enLista, carga1, carga5, entre otros.

Estos valores son comparados con los umbrales definidos por Hewlett Packard para maximizar su rendimiento, mismos que fueron recopilados a lo largo de varios años en el mantenimiento y afinación de sus equipos con sistema operativo hp-ux, que como sabemos, es uno de los más utilizados de los diferentes sabores existentes de Unix.

Si se tienen más de cuatro procesos en espera de ser ejecutados o si se llegan a tener más de 90 procesos en espera de ser atendidos, se estará presentando un cuello de botella en la cola de procesos.

Opciones myalix

Una vez seleccionada la opción K (Kolas) desde el menú principal de myalix, la primera línea del área de menús indicará que se está monitoreando la cola de procesos existente en el equipo.

La segunda línea de ésta área de menús, mostrará las siguientes opciones que podrán ser seleccionadas:

enEspera - Al teclear E la información desplegada será ordenada descendentemente por el número de procesos que se encuentran en espera de ser ejecutados en el sistema.

enLista - Al teclear L la información desplegada será ordenada descendentemente por el número de procesos que se encuentran en lista por ser atendidos en el sistema.

Carga1 - Al teclear 1 la información desplegada será ordenada descendentemente por porcentaje de carga de trabajo presentado en el equipo en el último minuto.

Carga5 **-** Al teclear 5 la información desplegada será ordenada descendentemente por porcentaje de carga de trabajo presentado en el equipo en los últimos cinco minutos.

umBrales - Al teclear B podremos consultar y modificar los umbrales de funcionamiento definidos para el disco. Una vez modificados se actualizarán dichos umbrales en el área de mensajes al final de la pantalla.

Regresar - Al teclear R, regresaremos al menú principal de la aplicación, para poder monitorear otro elemento.

Si myalix detecta que alguno de los valores han alcanzado o superado los umbrales definidos para su correcto funcionamiento, inmediatamente enviará un mensaje indicando que se está presentando un cuello de botella en la cola de procesos existentes, mostrará una bitácora de eventos realzando aquellos que han rebasado los umbrales definidos y recomendará las acciones a realizar con el propósito de poder afinarlo.

Siempre que los umbrales sean superados, myalix creará una bitácora que será alojado en el directorio correspondiente para ser consultado posteriormente por el administrador del equipo.

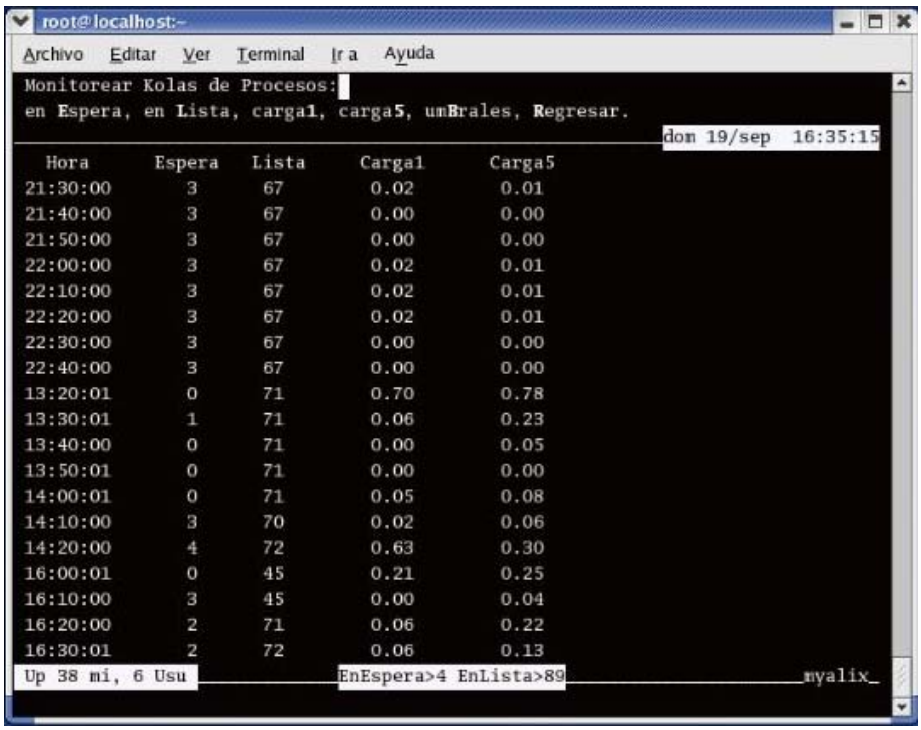
Programa MyAKol

Está realizado en programación *shell*, invoca al menú de opciones *MenuKol* escrito en lenguaje C y se encuentra localizado en el directorio monitor, dentro del disco compacto entregado con el presente trabajo.

Tiene una estructura similar del programa *MyACpu*, mostrado en éste capítulo.

Ejecución del Módulo

Se invoca con la letra K (Kolas) desde el menú principal y presenta la siguiente pantalla:



Cuando los valores mostrados rebasan los umbrales definidos en la parte inferior de la pantalla, el sistema informa al usuario del suceso, crea la bitácora de eventos e indica las acciones a realizar con el fin de poder afinar de la mejor forma posible el estado actual de la cola de procesos, tal y como lo ejemplificamos en el módulo MyACpu.

6. - MyAProc

Es el módulo encargado de monitorear y recomendar acciones para afinar los diferentes procesos existentes en el equipo que está siendo monitoreado, si los umbrales definidos para su desempeño han sido alcanzados o superados.

Es invocado al teclear la letra P (letra mayúscula de Procesos), desde el menú principal del sistema al preguntar por el recurso que se desea monitorear.

Para poder realizarlo, verificamos el estado de los procesos que se están presentes en el sistema, ocupando rutinas de administración propias de Linux, mismas que mostramos en el anexo del presente documento y las cuales ofrecen la siguiente información:

- Identificador del proceso asignado por el sistema operativo (pid).
- Comando que está asociado con el proceso (command).
- Usuario que disparó el comando (user).
- Prioridad asignada por el sistema operativo al proceso (pri).
- Porcentaje de tiempo dedicado por el procesador a la ejecución del proceso (%cpu).
- Terminal en la que se está ejecutando el comando (tty).

Umbrales de desempeño.

Para conocer el estado de cada uno de los procesos, se hace uso de los comandos de administración tales como ps, free y sar, que se detallan al final de éste capitulo, y de los cuáles se obtienen los valores pid, user, pri, nice, size entre otros.

Estos valores son comparados con los umbrales definidos por Hewlett Packard para maximizar su rendimiento, mismos que fueron recopilados a lo largo de varios años en el mantenimiento y afinación de sus equipos con sistema operativo hp-ux, que como sabemos, es uno de los más utilizados de los diferentes sabores existentes de Unix.

Si el número máximo de procesos es mayor a 99 o el tiempo máximo dedicado por el CPU es mayor al 20% o si alguno de los procesos que se están ejecutando tiene una prioridad mayor a 9, se estará presentando un cuello de botella en el sistema causado por los procesos que se están ejecutando.

Los umbrales óptimos de funcionamiento se muestran en la sección de mensajes en la parte inferior de la pantalla.

Opciones myalix

Una vez seleccionada la opción P (Procesos) desde el menú principal de myalix, la primera línea del área de menús indicará que se está monitoreando los procesos existentes en el equipo.

La segunda línea de ésta área de menús, mostrará las siguientes opciones que podrán ser seleccionadas:

Siguientes - Con la letra S se mostrarán descendentemente la siguiente lista de procesos existentes en el equipo, ordenándola por el número de identificación del proceso, conocido como *process id*.

Ultimos - Con la letra U se mostrarán descendentemente la última parte de la lista de procesos existentes en el equipo, ordenándola por el número de identificación del proceso, conocido como *process id*.

Etime - Con la letra E se ordenará descendentemente la lista de procesos por el *elapsed time* de cada uno de los procesos existentes en el sistema.

Cpu **-** Con la letra C se ordenará descendentemente la lista de procesos por el porcentaje de uso del procesador dedicado a cada uno de ellos.

umBrales - Al teclear B podremos consultar y modificar los umbrales de funcionamiento definidos. Una vez modificados se actualizarán dichos umbrales en el área de mensajes al final de la pantalla.

Regresar - Al teclear R, regresaremos al menú principal de la aplicación, para poder monitorear otro elemento.

Si myalix detecta que alguno de los valores han alcanzado o superado los umbrales definidos para su correcto funcionamiento, inmediatamente enviará un mensaje indicando que se está presentando un cuello de botella en los procesos existentes, mostrará una bitácora de eventos realzando aquellos que han rebasado los umbrales definidos y recomendará las acciones a realizar con el propósito de poder afinarlo.

Siempre que los umbrales sean superados, myalix creará una bitácora que será alojado en el directorio correspondiente para ser consultado posteriormente por el administrador del equipo.

Cada vez que se presenten problemas con algunos procesos del sistema, el administrador del equipo puede realizar alguna de las siguientes acciones:

- Disminuir la prioridad asignada a los procesos, evitando así que el procesador dedique demasiado tiempo en su atención

- Enviar la ejecución del proceso a back ground
- Eliminarlos de la sesión

Cualquiera de estas acciones deberán ser cotejadas con el administrador del sistema para evitar tocar procesos vitales que requiere Linux para su correcto funcionamiento.

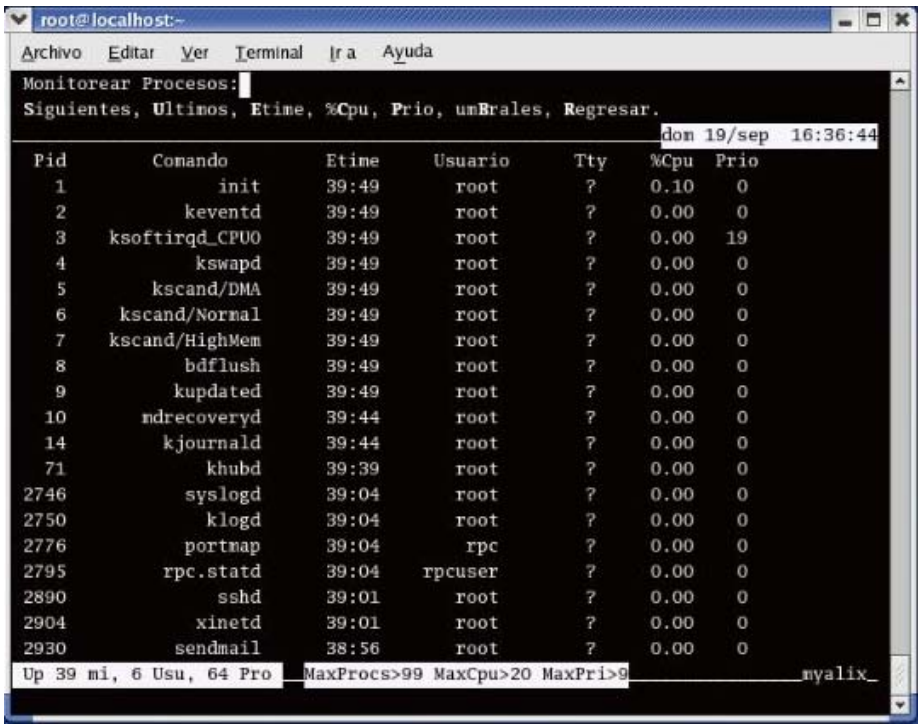
Programa MyAProc

Está realizado en programación *shell*, invoca al menú de opciones *MenuPro* escrito en lenguaje C y se encuentra localizado en el directorio monitor, dentro del disco compacto entregado con el presente trabajo.

Tiene una estructura similar del programa *MyACpu*, mostrado en éste capítulo.

Ejecución del Módulo

Se invoca con la letra P (Procesos) desde el menú principal y presenta la siguiente pantalla:



Cuando los valores mostrados rebasan los umbrales definidos en la parte inferior de la pantalla, el sistema informa al usuario del suceso, crea la bitácora de eventos e indica las acciones a realizar con el fin de poder afinar de la mejor forma posible el estado actual de los procesos, tal y como lo ejemplificamos en el módulo MyACpu.

Lo importante no radica en nunca caer,
sino en levantarse cada vez que caigamos.
Confucio

8. Conclusiones

Largo ha sido el camino que hemos recorrido desde que se vislumbró la idea de generar un sistema que fuera capaz de monitorear, y sobretodo, pudiera recomendar acciones con la única idea de poder afinar los recursos principales que componen el sistema operativo Linux, ya que las herramientas y comandos con las que cuenta, no sugieren acción alguna a tomar.

Creemos firmemente que logramos nuestro cometido, generando un sistema capaz de monitorear y recomendar acciones con el fin de afinar el rendimiento del sistema operativo Linux, lo que nos permitió adentrarnos de forma sin igual en este re encarnado mundo Unix, que parecía estar completamente olvidado y que probablemente las nuevas generaciones lo hubieran conocido solamente en los libros, mismos que en la actualidad están siendo amenazados por el creciente uso de Internet, pero que sin duda nunca desaparecerán.

Deseamos que este sistema, y similares, despierten en la gran cantidad de usuarios de computadoras personales el gusto por revisar qué es lo que pasa dentro de sus “gabinetes blancos”, dejando atrás el mito de las “cajas negras”; en lugar de considerar que tipo de memoria utilizar (*sims* o *dims)* o que nuevo procesador central se deberá adquirir para ampliar la capacidad de cómputo de sus equipos, deberán detectar que es lo que realmente se requiere para incrementar el desempeño de los mismos.

En países latinos como el nuestro, no podemos darnos el lujo de cambiar nuestras computadoras personales cada vez que aparece un nuevo modelo, por lo que el continuo monitoreo y afinación de nuestros equipos deberá ser una acción perenne a realizar, antes de tomar la difícil decisión de migrar todas las aplicaciones residentes a un nuevo hardware.

Es precisamente en éste fértil terreno en donde nuestro sistema de monitoreo y afinación del rendimiento del sistema operativo Linux, puede ayudar a realizar esta vital tarea de una forma sencilla, ágil, rápida y oportuna, evitando así que se presenten problemas irreversibles que no han sido aún detectados.

Comandos Linux

Para la estructura de Directorios.

cd	Cambia el directorio actual.
chgrp	Cambia el grupo propietario de archivos y directorios.
chmod	Cambia los permisos o modo de archivos y directorios.
chown	Cambia el usuario propietario de archivos y directorios.
cp	Copia archivos y directorios.
df	Muestra la cantidad de espacio disponible en disco.
du	Da información sobre el espacio ocupado en disco por archivos y directorios.
echo	Muestra una línea de texto en pantalla y permite conocer el valor de las variables de ambiente definidas.
file	Proporciona información sobre el tipo de un archivo.
find	Busca archivos y directorios.
ln	Crea enlaces o ligaduras a archivos.
ls	Lista archivos contenidos en un directorio.
mkdir	Crea directorios.
mv	Mueve archivos y directorios.
pwd	Muestra la ruta del directorio actual.
rm	Borra archivos y directorios.

Para el tratamiento de Archivos.

cat	Concatena archivos.
compress	Comprime archivos.
cut	Extrae una parte de cada línea de un archivo.
diff	Busca diferencias entre archivos.
ed	Editor de líneas.
grez	Busca patrones dentro de un archivo.
gzip	Compresor / descompresor de archivos.
head	Muestra las primeras líneas de un archivo.
more	Muestra el archivo por páginas.
paste	Produce su salida a base de columnas formadas por varios archivos.
tail	Muestra el final de un archivo.
tar	Crea archivos en cinta y genera copias de seguridad.
touch	Crea un archivo vacío.
uniq	Busca líneas iguales.
Vi	Editor de pantalla.

Para la gestión de Procesos.

at, batch	Programa la ejecución de procesos a una hora determinada, cambia al directorio actual una vez terminados.
crontab	Facilita la ejecución del demonio *cron* con los programas que están definidos para ejecutarse.
env	Ejecuta un comando o programa con el ambiente cambiado y permite ver el ambiente actual.
id	Muestra información sobre la identificación del usuario dentro del sistema.
kill	Comando para eliminar procesos ejecutándose en el sistema.
ps	Muestra la lista de procesos ejecutándose en ese momento en el sistema y su estado.
reboot	Reinicia el sistema.
shutdown	Da de baja el sistema.
sync	Escribe el contenido de los buffers en disco.
time	Informa sobre el tiempo de proceso que requiere el comando dado como parámetro.

Para propósito general.

banner	Muestra la cadena con los caracteres agrandados. Se utiliza para generar carteles.
cal	Muestra un calendario.
clear	Borra la pantalla.
date	Muestra hora y fecha del sistema.
finger	Proporciona información sobre los usuarios del sistema.
logname	Muestra el nombre con el que se inició la sesión en el sistema.
man	Muestra la ayuda de Linux.
mesg	Habilita la escritura de mensajes en la terminal.
passwd	Cambia la contraseña de entrada.
rpm	Comando utilizado para la instalación de paquetes RedHat en el sistema.
which	Busca programas.
who	Informa sobre los usuarios presentes en el sistema.
write	Permite escribir en una terminal.

Para la estructura de Directorios.

cd: *Cambia el directorio actual*

Sintaxis: cd otro_directorio

Descripción:
El comando cd cambia del directorio actual al indicado en otro_directorio. Si no se indica ningún directorio cambiará al *home directory* del usuario. Admite los directorios . y .. correspondientes al actual y al padre del actual para así indicar las rutas relativas al directorio destino.

Ejemplo: cd ../ShellJobs

chgrp: *Cambia el grupo propietario de archivos y directorios.*

Sintaxis: chgrp [*opciones*] grupo archivos

Descripción:
Este comando permite el cambio de grupo propietario de un archivo o directorio.
Para cambiar la propiedad de grupo de un archivo o directorio hay que ser el propietario de dicho archivo o ser el usuario *root.*

Ejemplo: chgrp users MainMenu

chmod: *Cambia los permisos o modo de archivos y directorios.*

Sintaxis: chmod [*opciones*] modo archivos

Descripción:
Este comando permite el cambio de los permisos o modo de los archivos y directorios. Los permisos de un archivo o directorio permiten controlar el acceso a los mismos.
Existen tres niveles de permisos, los del propietario, del grupo y los demás usuarios del sistema. Dentro de cada nivel de permisos existen tres tipos de derechos: de lectura (r), de escritura (w) y de ejecución (x).

Ejemplo: chmod g+r,g+w MainMenu

chown: *Cambia el usuario propietario de archivos y directorios.*

Sintaxis: chown [*opciones*] usuario archivos

Descripción:
Este comando permite el cambio de usuario propietario de un archivo o directorio. Para cambiar el propietario de un archivo o directorio hay que ser el propietario de dicho archivo o ser el usuario *root*.

Ejemplo: chown root MainMenu

cp: *Copia archivos y directorios.*

Sintaxis: cp [*opciones*] fuente destino

Descripción:
Mediante *cp* se copian los archivos y directorios. Copiará fuente a destino, fuente puede ser una lista de archivos que copiará a un directorio indicado en destino. También puede copiar un archivo sobre otro archivo.

Ejemplo: cp MainMenu ../ShellJobs

df: *Muestra la cantidad de espacio disponible en disco.*

Sintaxis: df [*opciones*] [sistema_de_archivos]

Descripción:
Este comando muestra la cantidad de espacio que queda libre en disco para un sistema de archivos o partición dado.

Ejemplo: df

du: *Da información sobre el espacio ocupado en disco por archivos y directorios.*

Sintaxis: du [*opciones*] archivos

Descripción:
Informa sobre la cantidad de espacio que ocupan en disco los archivos y directorios indicados como parámetros. Si no da ningún parámetro al comando, el resultado del comando será la información del directorio actual.

Ejemplo: du

echo: *Muestra una línea de texto en pantalla y permite conocer el valor de las variables de ambiente definidas.*

Sintaxis: echo [-n] [-e] cadena

Descripción:
Este comando permite mostrar en la salida estándar la cadena que se le pasa como argumento. Su utilidad está en los shell scripts para dar información.

Ejemplo: echo –n "¿ Desea continuar con el proceso ? "

file: *Proporciona información sobre el tipo de un archivo.*

Sintaxis: file [opciones] archivos

Descripción:
Este comando nos da información sobre un archivo referente al tipo del mismo. Así nos dará información sobre si un archivo es ejecutable, de datos o de texto.

Ejemplo: file /etc/passwd

find: *Busca archivos y directorios.*

Sintaxis: find [ruta] [expresión]

Descripción:
Con este comando podemos realizar todo tipo de búsquedas de archivos y directorios en el sistema de archivos, atendiendo a una serie de características y patrones de los archivos y directorios que se desean localizar.
Es posible realizar búsquedas atendiendo a criterios especificados en expresión evaluándolo de izquierda a derecha. También nos permite ejecutar un comando a cada archivo localizado. Por defecto se toma el directorio actual para iniciar la búsqueda y la opción *–print* como expresión.

Ejemplo: find / -name MainMenu -print

ln: *Crea enlaces o ligaduras a archivos.*

Sintaxis: ln [-s] origen destino

Descripción:
Comando que crea un enlace a un archivo, con el fin de poder acceder a un archivo con más de un nombre. Se pueden utilizar estos enlaces para copiar archivos sin ocupar espacio en disco. Podemos tener dos tipos de enlaces, los simbólicos y los fijos. Los enlaces fijos o ligaduras físicas crean una nueva entrada en el directorio pero no crean un nuevo inodo, sino que a esa nueva entrada se le asigna el número de inodo origen. Las ligaduras simbólicas crean un archivo especial cuyo contenido es origen.

Ejemplo: ln MainMenu ../comandos/ShellM

ls: *Lista archivos contenidos en un directorio.*

Sintaxis: ls [opciones] archivos

Descripción:
Lista los archivos indicados mostrando información sobre ellos. Si no se para como parámetro ningún nombre de archivo, lista los archivos del directorios actual.

Ejemplo: ls -al

mkdir: *Crea directorios.*

Sintaxis: mkdir [opciones] directorio

Descripción:
Comando utilizado para la creación de directorios.

Ejemplo: mkdir Cjobs

mv: *Mueve archivos y directorios.*

Sintaxis: mv [opciones] origen destino

Descripción:
Comando utilizado para mover archivos y directorios. El comando en sí lo que realiza es la copia del origen al destino y el borrado del origen. También es el comando que se utiliza para renombrar archivos y directorios por ser similar el efecto que produce.

Ejemplo: mv OldName NewName

pwd: *Muestra la ruta del directorio actual.*

Sintaxis: pwd

Descripción:
Este comando informa sobre cuál es la ruta en la que nos encontramos dentro del sistema.

Ejemplo: pwd

rm: *Borra archivos y directorios.*

Sintaxis: rm [opciones] archivos

Descripción:
Elimina archivos y directorios del sistema de archivos. Se utiliza tanto para borrar archivos, enlaces y directorios. Para eliminar directorios es necesaria la opción *–r*, pero se debe tener mucho cuidado con el uso de este comando, ya que sus acciones son irreparables.

Ejemplo: rm –r *

Para el tratamiento de Archivos.

cat: *Concatena archivos.*

Sintaxis: cat [opciones] archivos

Descripción:
Concatena la lista de archivos dados como parámetro con la salida estándar. Si sólo se le indica un archivo, lo muestra. Si se le indica, lo crea con el contenido que se escriba en la entrada estándar finalizando con <ctrl> <D> para grabar o <ctrl> <C> para salir sin grabar.

Ejemplo: cat MainMenu

compress: *Comprime archivos.*

Sintaxis: compress [opciones] archivos

Descripción:
Comprime los archivos que se le indican, también sirve para dar el paso contrario y descomprimir los archivos previamente comprimidos. La compresión de un archivo hace que a su nombre se la añada .z .

Ejemplo: compress MainMenu

cut: *Extrae una parte de cada línea de un archivo.*

Sintaxis: cut [opciones] archivos

Descripción:
Se utiliza para extraer de un archivo determinadas columnas en forma de campos. Estos campos se definen por medio de separadores, o indicando el número de carácter explícitamente o por medio de un rango.

Ejemplo: cut –f1 –d: /etc/passwd

diff: *Busca diferencias entre archivos.*

Sintaxis: diff [opciones] archivo1 archivo2

Descripción:
Encuentra las diferencias entre dos archivos de texto, indicando lo que tendríamos que hacer para que coincidieran. Se le puede dar como parámetros nombres de directorios, así se da como primer nombre de archivo un directorio y como segundo un archivo, compara el archivo del directorio cuyo nombre coincida con el nombre del segundo archivo dado. El caso contrario funciona igual. Si se dan dos nombres de directorios comparara por orden alfabético archivos de igual nombre.
Ejemplo: diff capitulo1.txt capitulo1.bak

ed: *Editor de líneas.*

Sintaxis: ed [opciones] archivo

Descripción:
Editor de líneas que se suele utilizar cuando no es posible el uso de un editor de pantalla completa. Es un editor bastante complejo por lo que, para más información, será necesario consultar el manual específico del editor *ed*.

Ejemplo: ed Capitulo1.bak

grep: *Busca patrones dentro de un archivo.*

Sintaxis: grep [opciones] expresión_regular archivos

Descripción:
Busca expresiones regulares que se le pasan como parámetros dentro de archivos, dando como salida donde aparecen dentro del archivo dichas expresiones. Así este comando se puede utilizar para la búsqueda de archivos que contengan determinadas cadenas de caracteres o para la búsqueda dentro de archivos de determinadas expresiones.

Ejemplo: grep “include” h*

gzip: *Compresor / descompresor de archivos.*

Sintaxis: gzip [opciones] archivos

Descripción:
Es un compresor / descompresor de archivos, reduce el tamaño de los archivos mediante un algoritmo de compresión. Genera archivos con extensión .gz, por omisión si se comprime un archivo gzip lo reemplaza por otro ya comprimido manteniendo sus permisos y derechos. Este comando tiene su complementario, aunque realmente no hace falta, que es el comando unzip que descomprime un archivo, pero gzip es capaz de devolver los archivos a su estado original. Generalmente se utiliza en combinación con el comando tar.

Ejemplo: gzip *

head: *Muestra las primeras líneas de un archivo.*

Sintaxis: head –líneas archivos

Descripción:
Muestra las primeras líneas de los archivos dados.

Ejemplo: head -9 .profile

more: *Muestra el archivo por páginas.*

Sintaxis: more [opciones] archivos

Descripción:
Se utiliza para paginar texto que no cabe en la pantalla y, por tanto, se desplaza sin poder verlo. Se utiliza a través de un redireccionamiento o un *pipe* para formatear el resultado de otro comando como *cat* o *ls*.

Ejemplo: ls –l more

paste: *Produce su salida a base de columnas formadas por varios archivos.*

Sintaxis: paste [opciones] archivos

Descripción:
Genera en su salida varias columnas formateadas por las líneas de varios archivos, su utilidad radica en la posibilidad de unir varios archivos en uno solo, redireccionando la salida del comando, que estaría formado por tantas columnas como archivos procesa.

Ejemplo: paste Cap1 Cap2

tail: *Muestra el final de un archivo.*

Sintaxis: tail [opciones] archivo

Descripción:
Tiene un funcionamiento similar al comando head, pero en sentido contrario, ya que muestra las últimas líneas del archivo que se le pasa como parámetro. Por omisión muestra las 10 últimas líneas.

Ejemplo: tail MainMenu

tar: *Crea archivos en cinta y genera copias de seguridad.*

Sintaxis: tar accion [opciones] archivos

Descripción:
La función fundamental de este comando es la de realizar copias de archivos hacia un dispositivo de cinta. Pero también se utiliza el comando para generar copias de seguridad de varios archivos en uno solo. Muchos paquetes y programas vienen en archivos *tar* por su comodidad de distribución.

Ejemplo: tar cvf respaldo.tar

touch: *Crea un archivo vacío.*

Sintaxis: touch archivo

Descripción:
Permite crear un archivo con longitud de cero bytes, el cuál es muy útil para inicializar *raw devices* en la instalación de manejadores de bases de datos.

Ejemplo: touch ArchLongCero

uniq: *Busca líneas iguales.*

Sintaxis: uniq [opciones] entrada salida

Descripción:
Toma la entrada y la procesa para sacar una sola vez cada línea de entrada. Así si hay más de dos líneas iguales sólo muestra una de ellas.
Ejemplo: uniq VariasVeces UnaSolaVez

vi: *Editor de pantalla.*

Sintaxis: vi archivos

Descripción:
Permite crear y modificar archivos de texto tanto de uso personal como del sistema. Linux tiene definida toda su configuración en archivos tipo texto, así que es un comando muy útil, pero a la vez no muy amigable en su uso.

Ejemplo: vi /etc/passwd

Para la gestión de Procesos.

at, batch: *Programa la ejecución de procesos, cambia al directorio actual.*

Sintaxis: at [- opciones] [cola] hora [fecha] [incremento] trabajo
Batch trabajo

Descripción:
Tanto el comando at se utiliza para programar la ejecución de trabajos a una determinada hora, el comando batch lo que hace es ejecutar los comandos cuando el nivel de carga del sistema es bajo.
Así, se puede utilizar para programar la ejecución de tareas de baja prioridad cuando el sistema tiene menos actividad. Tanto at como batch el terminar la ejecución de un trabajo envían por defecto la salida al correo del usuario. Los trabajos planificados con el comando bach pueden ser cancelados antes de que comience su ejecución.

Ejemplo: at 17:00 RespaldaArchivos

crontab: *Facilita la información del demonio cron de los programas que hay programados para ejecutarse.*

Sintaxis: crontab [opciones] [-u usuario] [archivo]

Descripción:
Por medio del comando *crontab* podemos planificar la ejecución de comandos con cierta periodicidad. Este comando utiliza del demonio *cron* que debe estar en ejecución si queremos que funcione correctamente el comando *crontab*. Los usuarios crearán un archivo con las órdenes que se quieren planificar con *crontab*, y ejecutarán el comando con el archivo como argumento para fijar ese como su archivo *crontab*. Solamente el usuario *root* puede manejar los archivos *crontab* de cada usuario que se encuentran en el directorio /usr/spool/cron/crontabs, donde hay uno por usuario.

Ejemplo: 0 * * * * sync

env: *Ejecuta un comando o programa con el ambiente cambiado y permite ver el ambiente actual.*

Sintaxis: env [opciones] comando

Descripción:
Permite cambiar el ambiente de ejecución, variables de ambiente, para la ejecución de un comando o programa sin variar el ambiente actual. Los cambios de las variables de ambiente sólo afectarán al comando que se quiere lanzar con *env*. Si lo ejecutamos sin ningún parámetro sirve para ver el ambiente actual.

Ejemplo: env HOME=/root/shells cd

id: *Muestra información sobre la identificación dentro del sistema.*

Sintaxis: id [opciones]

Descripción:
Muestra información sobre la identificación del usuario dentro del sistema, que esta ejecutando el comando. La información que muestra consiste en el nombre del usuario: UID y nombre del grupo al que pertenece: GID.

Ejemplo: id

Kill: *Comando para eliminar procesos ejecutándose en el sistema.*

Sintaxis: kill [-señal] pid

Descripción:
Comando utilizado generalmente para detener procesos que se están ejecutando, comúnmente se llama matar a un proceso a esta acción. En general, la acción que se realiza con el comando *kill* es mandar señales a los procesos en ejecución. Solamente podemos matar el proceso si se es el usuario que lanzó dicho proceso o si se es el usuario *root*.

Ejemplo: kill -9 1321

ps: *Muestra la lista de procesos ejecutándose en el sistema y su estado.*

Sintaxis: ps [opciones]

Descripción:
Este comando se utiliza para ver en un instante dado la ejecución de los comandos y el estado de los procesos que se están ejecutando en el sistema. Si no se especifica ninguna opción, da información sobre los procesos del intérprete de comandos en el que se ejecuta. Este comando es muy útil para averiguar el pid de procesos de que quieren matar con el comando kill. La información que facilita el comando ps sobre los procesos, se presenta en formato de columnas y dependiendo de las opciones nos muestra la siguiente información en sus correspondientes columnas:

ADDR: Dirección de memoria del proceso en ejecución.
F: Flags del proceso
- 1: Proceso trazado
- 2: El proceso está asignado al procesador
- 3: El proceso no está asignado

PID: Identificador del proceso
PPID: Identificador del proceso padre que lo creó
PRI: Prioridad del proceso
SIZE: Espacio en número de páginas que ocupa el proceso en memoria.
RSS: Tamaño del proceso en kilobytes que está realmente en memoria en ese instante
WCHAN: Evento sobre el que el proceso está esperando
S: Estado del proceso:

-: El proceso no existe
S: Durmiendo
W: En espera de algún evento
R: En ejecución
I: Recién creado
Z: Zombie
T: Trazado

TTY: Terminal del proceso
PAGEIN: Cantidad de faltas de páginas que ha causado
TIME: Tiempo de CPU consumido por el proceso
UID: Identificador del usuario que lanzó el proceso
SWAP: Cantidad de kilobytes del proceso en la memoria SWAP.

Ejemplo: ps

reboot: *Reinicia el sistema.*

Sintaxis: reboot

Descripción:
Este comando funciona como un *shutdown*, pero lo hace de forma inmediata y sin parámetros. Justo cuando se lanza, sólo por el usuario *root* y usuarios autorizados, comienza la secuencia de cierre y rearranque del sistema. La combinación de teclas [CTRL] [ALT] [SUP] tiene el mismo efecto.

Ejemplo: reboot

shutdown: *Da de baja el sistema.*

Sintaxis: shutdown [opciones] hora [mensaje]

Descripción:
Este comando se utiliza para apagar y reiniciar el sistema. Linux es un sistema operativo que no se puede apagar sin más, dado que esto podría causar daños irreparables. Antes de apagar una computadora con Linux hay que guardar en disco una serie de datos y parámetros que están en memoria así como finalizar una serie de procesos. También, ya que se trata de un sistema multiusuario, si hay más de un usuario en el sistema sería conveniente avisarles para que guarden sus datos antes de apagar o reiniciar la computadora. Por eso es necesario este comando que realiza todas esas tareas y deja la computadora lista para ser apagado. Este comando solamente lo puede ejecutar el usuario *root*, y usuarios autorizados.

Ejemplo: shutdown

sync: *Escribe los buffers en disco.*

Sintaxis: sync

Descripción:
La acción de este comando es la de escribir en disco los *buffers* de E/S del sistema de archivos. Si por ejemplo se copia un archivo en el disco y apagamos súbitamente el sistema, al reiniciar podrá ocurrir que el archivo no sea encontrado. Esto es porque realmente cuando realizamos operaciones sobre el sistema de archivos en realidad estamos trabajando con *buffers* en memoria para acelerar las operaciones. El efecto del comando *sync* es vaciar los *buffers* de memoria y escribir su contenido en disco definitivamente.

Ejemplo: sync

time: *Informa sobre el tiempo de proceso de un comando.*

Sintaxis: time comando

Descripción:
Este comando facilita la información de tiempo que consume un comando. La información que nos facilita es de tres tipos: tiempo real que transcurre desde que se lanza el comando hasta que termina, tiempo de usuario que es el tiempo que realmente ha utilizado de CPU y, por último, tiempo de sistema que es el tiempo consumido en peticiones realizadas al sistema.

Ejemplo: time MainMenu

Para propósito general.

banner: *Muestra la cadena con los caracteres agrandados. Se utiliza para generar carteles.*

Sintaxis: banner [[-w] [n]] cadena_de_caracteres

Descripción:
Genera como salida la cadena de caracteres que se le pasa como entrada y en forma de letrero, forma las letras a base de asteriscos.

Ejemplo: banner BIENVENIDO

cal: *Muestra un calendario.*

Sintaxis: cal [-j] [-y] [mes] [año]

Descripción:
Este comando muestra en pantalla el calendario del año indicado, o del mes indicado.

Ejemplo: cal 5 2004

clear: *Borra la pantalla.*

Sintaxis: clear

Descripción:
Borra la pantalla completamente de la Terminal en la que se está trabajando.

Ejemplo: clear

date: *Muestra hora y fecha del sistema.*

Sintaxis: date [MMDDhhmm] [CC] [AA] [.ss] [+formato]

Descripción:
Este comando permite ver la hora y la fecha del sistema. Si se es el usuario root además permite cambiar día, hora y fecha. El formato de salida tiene multitud de opciones y se recomienda consultar la página mas correspondiente para verlas todas. Para establecer la hora está se indica don el formato:

MMDDhhmm [CC] [AA] [.ss]

Ejemplo: date

finger: *Proporciona información sobre los usuarios del sistema.*

Sintaxis: finger [opciones] usuarios

Descripción:
Este comando se utiliza para obtener información sobre los usuarios del sistema, muestra toda la información referente a un usuario que este registrado en el sistema. Si sólo se invoca el comando sin ninguna opción ni nombre de usuario, nos da información referente a los usuarios que actualmente se encuentran conectados al sistema. Podemos pedir información sobre los usuarios de otras computadoras conectadas en red utilizando:

usuario@nombre_dela_computadora.sudominio

Ejemplo: finger mahmc

logname: *Muestra el nombre con el que se inició la sesión en el sistema.*

Sintaxis: logname

Descripción:
Muestra el nombre de usuario con el que se entró en el sistema que lo invoca.

Ejemplo: logname

man: *Muestra la ayuda de Linux.*

Sintaxis: man [nivel] comando

Descripción:
Este es quizá el comando más importante de Linux, porque es el que nos facilita la ayuda del mismo. Invocando este comando con el comando o función sobre el cuál queremos información como parámetro, nos muestra una descripción bastante completa y muy útil. Las páginas *man*, que es como se llama la información que se muestra en pantalla, están divididas en niveles, en ocasiones algunos comandos vienen en varios niveles y el comando *man* muestra la primera página que encuentre, por eso se puede especificar el nivel al que nos referimos para distinguir entre comandos y funciones con el mismo nombre.

Ejemplo: man man

mesg: *Habilita la escritura de mensajes en la terminal.*

Sintaxis: mesg [opciones]

Descripción:
Habilita la escritura en la Terminal a otros usuarios mediante el comando write como sistema sencillo de mensajería.
Ejemplo: mesg y

passwd: *Cambia la contraseña de entrada.*

Sintaxis: passwd [nombre_del_usuario]

Descripción:
Mediante este comando cambiamos la contraseña de entrada al sistema. Salvo el usuario *root* sólo es posible cambiar la contraseña propia para lo cual no es necesario pasar el nombre de usuario como parámetro del comando. El usuario *root* puede cambiar la contraseña a cualquier usuario.

Ejemplo: passwd mahmc

rpm: *Comando utilizado para la instalación de paquetes RedHat en el sistema.*

Sintaxis: rpm [opciones] paquete

Descripción:
El comando rpm se utiliza para la instalación de paquetes en el formato rpm, que es el formato de los paquetes RedHat, aunque se está convirtiendo en un estándar de facto para la distribución de programas Linux. Con este comando además de instalar, también podemos desinstalar y actualizar cualquier paquete. Se ofrece también la posibilidad de crear paquetes.

Ejemplo: rpm –i sybase.1-1.23,rpm

which: *Busca programas.*

Sintaxis: which programa

Descripción:
Busca la existencia de un determinado programa en el sistema, la búsqueda la realiza solamente en la ruta de acceso del usuario que lo invoca.

Ejemplo: which MainMenu

who: *Informa sobre los usuarios presentes en el sistema.*

Sintaxis: who [opciones]

Descripción:
Informa sobre que usuarios están presentes en ese instante en el sistema.

Ejemplo: who

write: *Permite escribir en una terminal.*

Sintaxis: write usuario

Descripción:
Permite hacer uso de un sencillo sistema de mensajería donde se escribe directamente en la terminal del usuario que se indica, no hay que olvidarse que el usuario destinatario debe haber habilitado este servicio mediante el comado mesg.

Ejemplo: write mahmc

Para intercambio de información entre Linux y ms-dos.

mcd, mdir: *Permite cambiar de directorio en un disco formateado en ms-dos y mostrar su contenido.*

Sintaxis: mcd <directorio>
mdir <directorio>

Descripción:
Permite cambiar de directorio en un disco formateado en el sistema operativo ms-dos (cualquier Windows) y dar el listado de los archivos ahí almacenados.

Ejemplo: mcd a:\cjobs
mdir a:

mcopy: *Permite copiar archivos de Linux a un disco formateado en ms-dos y viceversa.*

Sintaxis: mcopy <archivos> <destino>

Descripción:
Permite copiar archivos generados en Linux a un disco formateado en el sistema operativo ms-dos (cualquier Windows) y del disco formateado en ms-dos a Linux.

Ejemplo: mcopy *.c a:\cjobs

mdel: *Permite borrar archivos de un disco formateado en ms-dos.*

Sintaxis: mdel <archivos>

Descripción:
Permite borrar archivos un disco formateado en el sistema operativo ms-dos (cualquier Windows).

Ejemplo: mdel a:\cjobs*.txt

Para la administración del sistema.

free: *Despliega información sobre la memoria libre y utilizada en el sistema.*

Sintaxis: free –[opciones]

Descripción:
Despliega la cantidad total de memoria física disponible y utilizada, así como en el área *swap* del sistema, también en los *buffers* y memoria *cache* utilizada por el kernel. Pueden utilizarse diferentes banderas al invocarlo.

Ejemplo: free -g

iostat: *Reporta el uso de procesador central, así como el uso de los dispositivos de entrada/salida.*

Sintaxis: iostat –[opciones]

Descripción:
Es utilizado para monitorear el uso de los dispositivos de entrada/salida existentes, así como el uso del procesador central. Puede invocarse con diferentes opciones.

Ejemplo: iostat –d 2 6

mpstat: *Genera estadísticas del uso del procesador.*

Sintaxis: mpstat –[opciones]

Descripción:
Escribe en la salida estándar, la actividad de cada uno de los procesadores existentes. En caso de que el sistema cuente con varios procesadores, el primero es numerado con 0 y los demás son numerados consecutivamente. También informa de las actividades globales generadas entre ellos.

Ejemplo: mpstat –P ALL 2 5

sar: *Colecciona, reporta y guarda la información de la actividad del sistema.*

Sintaxis: sar –[opciones]

Descripción:
Reporta en la salida estándar, los contadores seleccionados de la actividad acumulada del sistema operativo. El comando reporta las estadísticas que han sido almacenadas en el archivo: /var/log/sa/sa*dd*, donde *dd* es el número del día reportado.

sar son las siglas de *System Activity Report* y puede ser invocado con alguna de las siguientes opciones:

b: reporta estadísticas de entrada/salida.
B: muestra estadísticas de paginación.
c: reporta la actividad de creación de procesos.
d: informa la actividad de cada uno de los dispositivos de bloque.
e: define la hora de término del reporte.
f: obtiene valores de un archivo creado con la opción –o.
I: muestra estadísticas de una interrupción dada.
n: reporta estadísticas de uso de la red.
o: guarda el reporte en un archivo con formato binario.
q: informa la longitud de la cola de procesos y el promedio de carga.
r: reporta el uso de memoria y área de intercambio.
R: reporta el uso de la memoria principal.
s: fija la hora de inicio del reporte.
u: informa sobre el uso del procesador central.
U: reporta la actividad del procesador dado como parámetro, puede usarse ALL.
v: informa sobre el estado de los *inodos*.
V: imprime la versión utilizada de sar.
w: reporta la actividad de intercambio del sistema.
W: reporta las estadísticas de *swapeo*.
x: da información sobre el proceso dado.
X: da información sobre el proceso hijo del proceso dado.
y: reporta la actividad de las terminales del sistema.

Cada una de las anteriores opciones genera información específica sobre el estado del sistema, por lo que es conveniente tenerlo presente para poder hacer correcto uso del resultado del comando.

Ejemplo: sar –u 2 5

top: *Despliega los procesos más demandantes existentes en el sistema.*

Sintaxis: top

Descripción:
El comando genera una visión continua en tiempo real de los procesos existentes, mostrando las tareas más intensivas en el sistema y provee una interfase interactiva para manipular dichos procesos. No requiere de parámetros al invocarlo y para desactivarlo habrá que oprimir la letra <q>.

Ejemplo: top

uptime: *Informa cuanto tiempo ha estado el sistema en uso.*

Sintaxis: uptime

Descripción:
Despliega una línea con la siguiente información: tiempo que el sistema ha estado activo, cuantos usuarios se encuentran conectados y los promedios de carga del sistema en los pasados cinco y quince minutos. No requiere de parámetros al invocarlo.

Ejemplo: uptime

utmp: *Información sobre el registro de sesiones.*

Sintaxis: utmp

Descripción:
El archivo utmp nos permite obtener información de quienes están haciendo uso del sistema actualmente, pero puede haber más usuarios haciendo uso el sistema en el momento actual ya que no todos los programas usan utmp como registro de sesiones. No debe ser modificado ya que muchos programas del sistema dependen totalmente de su integridad. No requiere de parámetros al invocarlo.

Ejemplo: utmp

vmstat: *Genera el reporte de estadísticas de la memoria virtual.*

Sintaxis: vmstat [opciones]

Descripción:
Da información sobre el uso los procesos existentes, memoria, paginación, bloques de entrada/salida, trampas y actividad del procesador central. El primer informe producido da los promedios desde el último *reboot*.

Ejemplo: vmstat -n

w: *Muestra quien está haciendo uso del sistema y que es lo que realizan.*

Sintaxis: w

Descripción:
Muestra información de los usuarios que se encuentran actualmente en el sistema y el estado de sus procesos. El encabezado muestra en este orden, la hora actual, cuanto tiempo ha estado el sistema en ejecución, los usuarios que están conectados y el porcentaje de carga del sistema en los últimos cinco y quince minutos. No requiere de parámetros para su ejecución.

Ejemplo: w

Bibliografía

- Linux guía de instalación y administración
  Vicente López Camacho
  Osborne McGraw-Hill, 2001.

- HP-UX Series 800 performance and tuning
  HP Computer/instrument systems training course
  Hewlett Packard, 1992.

- UNIX Sistema V Version 4
  Kenneth H. Rosen
  Osborne McGraw Hill, 1991

- El entorno de programación UNIX
  Brian W. Kernighan
  Prentice Hall Hispanoamericana, 1987.

- Advanced UNIX programming
  Marc J. Rochkind
  Prentice Hall Software Series, 1985.

- Life with UNIX a guide for everyone
  Don Libes
  Prentice Hall, 1989.

- El sistema UNIX y sus aplicaciones
  José Canosa
  Publicaciones Marcombo, 1984.

- SCO UNIX, Operating System
  Tutorial
  Sco, open system software, 1992.

- Fundamentos del Sistema Operativo UNIX
  HP Computer Systems
  Hewlett Packard, 1993.