Materia: Introducción a los Sistemas Operativos / Sistemas Operativos
Tema: Sistema Operativo GNU/Linux
Versión: Agosto 2008
Palabras daves: GNU/Linux, Unix, Licencias, Linux, Kernel, Shell, Filesystem.

Los temas que se verán son:

1. - Introducción de UNIX
2. - El sistema GNU
3. - Conceptos de Linux
4. - Introducción breve de GNU-Linux
5. - Kernel (Linux)
6. - Shell
7. - Filesystem
8. - Comandos

Nota: Esta guía tiene como fin servir de apunte introductorio a algunos conceptos que son tratados en las prácticas de Sistemas Operativos Aplicados (SOA). Se trataran conceptos teóricos y prácticos; pero no se abordaran profundamente en cada uno de dichos conceptos. Si se desea seguir leyendo, al final de cada tema, se encuentran enlaces que amplían el contenido de esta guía.

1.- Introducción de UNIX

Unix es un sistema operativo creado a fines de 1969, por un grupo de investigadores de los laboratorios Bell de AT&T. A mediados de la década del 60, los investigadores de dicho laboratorios, en asociación con General Electrics, estaban desarrollando Multics que, hoy en día, se lo considera como uno de los primeros sistemas operativos de tiempo compartido. Multics trajo consigo nuevas ideas a nivel de software y hardware (-que sirvieron como influencia para la creación de posteriores sistemas operativos, entre ellos Unix-). A pesar de esto, dichas ideas fueron fuertemente criticadas, y Multics no alcanzo el éxito que se esperaba; por lo que los laboratorios Bell decidieron desvincularse y dedicar sus recursos a otros proyectos.

A pesar del fracaso de Multics, investigadores del laboratorio, comenzaron con una nueva creación de un sistema operativo, que tenía como filosofía de diseño, la idea de conseguir un sistema tan pequeño y simple como fuese posible; y de este modo corrigiendo lo que se veía como una deficiencia de Multics, lograron obtener un sistema operativo que tenia las características (-a pesar de ciertas limitaciones-), de ser portable, multitarea y multiusuario.

Originalmente el sistema fue bautizado como Unics (**Un**iplexed **I**nformation and **C**omputing **S**ystem), pero dada a la analogía hecha con el nombre del sistema Multics (**Mult**iplexed **I**nformation and **C**omputing **S**ervice) y sumado a que se decía que Unics era, prácticamente, una versión modificada del Multics, se decidió renombrarlo como Unix.

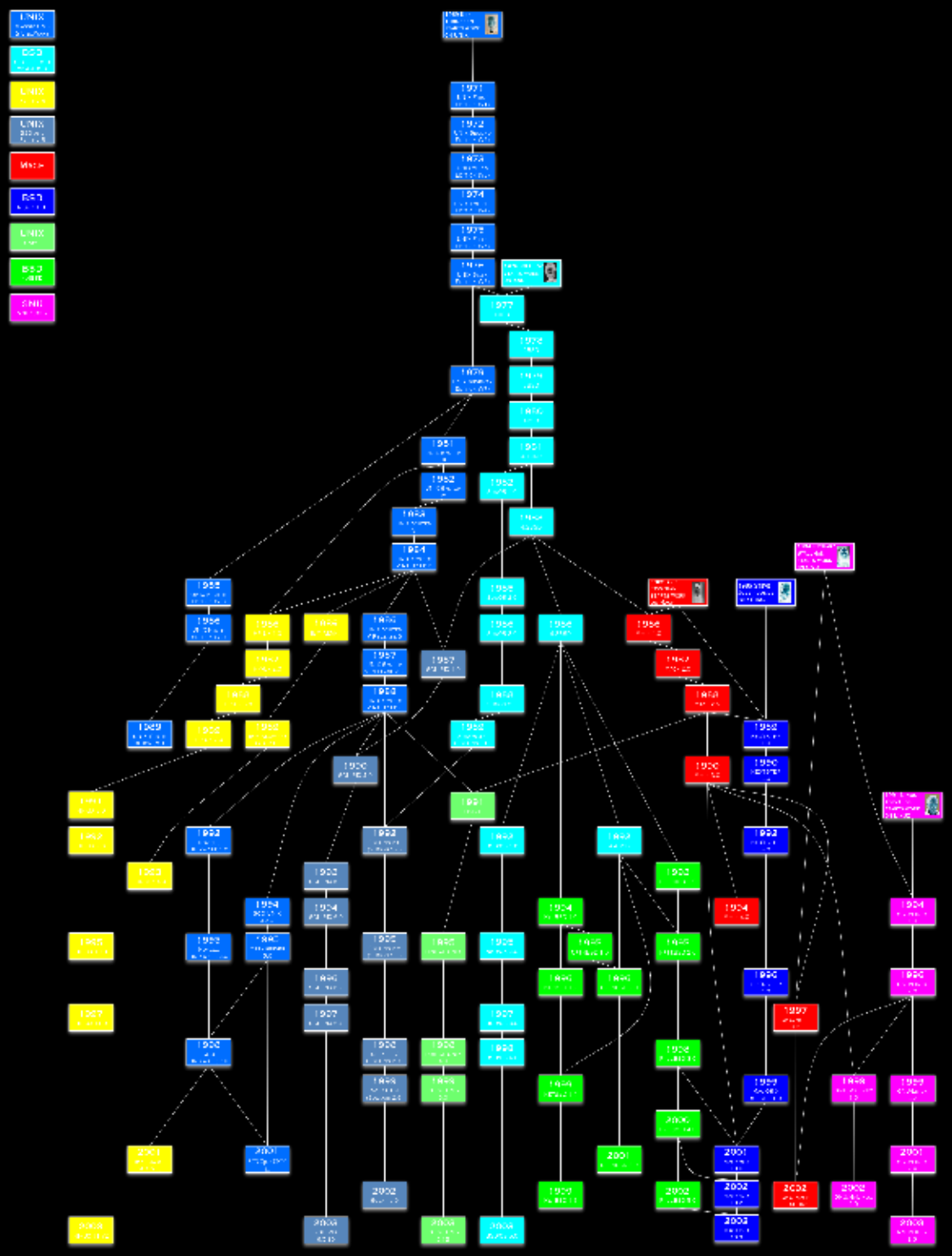
Entre los años 1969 y 1972, Unix tuvo un inesperado éxito, y ya comenzaba a hacerse popular, dado a que, como era de libre distribución, varios organismos comenzaron a utilizarlo; tal es así, que Unix consiguió el apoyo económico de los laboratorios Bell (-el mismo laboratorio que le había quitado el apoyo a la creación del sistema Multics-).

En 1972 se tomó la decisión de escribir nuevamente Unix, pero esta vez en el lenguaje de programación C. El nuevo código era mas conciso y compacto, lo que lo hacia que fuese mas fácil de modificarlo y adaptarlo a las necesidades de otros ordenadores.

Sobre finales de 1973 se toma la decisión de distribuir el sistema a las universidades, con la motivación de que Unix se hiciese aun mas popular; Y debido a que dicha motivación es alcanzada, fue a partir de ese momento que Unix tiene un cambio radical, ya que a lo largo de estos años comenzaron a crearse nuevas versiones del sistema.

Para ver cómo fueron apareciendo dichas versiones según cada fabricante, se puede consultar el sitio: http://www.levenez.com/unix/history.html#01, en donde se muestra la línea cronológica de las versiones del sistema, que van desde su creación hasta la actualidad.

Ó haciendo zoom sobre la siguiente imagen, se puede observar la cronología de las versiones, junto con el líder principal de cada proyecto.



El anterior cuadro, deja ver que el éxito de Unix en si mismo fue tan grande, que hoy en día es necesario diferenciar dicho término según el ambiente en donde es usado:

- La familia UNIX: desde el punto de vista técnico, UNIX se refiere a una familia de sistemas operativos que comparten unos criterios de diseño e interoperabilidad en común. Esta familia incluye más de 100 sistemas operativos desarrollados a lo largo de 20 años. No obstante, es importante señalar que esta definición no implica necesariamente que dichos sistemas operativos compartan código o cualquier propiedad intelectual.

- El sistema operativo original UNIX: desde el punto de vista histórico, UNIX se refiere a la subfamilia de sistemas operativos que descienden de la primera implementación original de AT&T. El término "descendencia" ha de interpretarse como trabajos derivativos que comparten propiedad intelectual con la implementación original.

- La marca UNIX: desde el punto de vista legal, Unix es una marca de mercado. Dicha marca es propiedad de "The Open Group", una organización de estandarización que permite el uso de dicha marca a cualquier sistema operativo que cumpla con sus estándares publicados (Single Unix Specification). Todo ello independientemente de que el sistema operativo en cuestión sea descendiente o clónico del Unix original. Resumiendo, la marca Unix no es propiedad de ninguna compañía.

Fuentes:

http://es.wikipedia.org/wiki/Unix
http://www.bell-labs.com/history/unix/
http://www.unix.org/

2.-El sistema GNU

El sistema GNU es el sistema operativo similar a Unix, constituido en su totalidad por software libre. Un sistema operativo similar a Unix está constituido por muchos programas. El sistema GNU incluye todo el software GNU, además de muchos otros paquetes, como el sistema X Window y TeX, los cuales no son software de GNU.

UNIX es un Sistema Operativo no libre muy popular, porque está basado en una arquitectura que ha demostrado ser técnicamente estable. El sistema GNU fue diseñado para ser totalmente compatible con UNIX. El hecho de ser compatible con la arquitectura de UNIX implica que GNU esté compuesto de pequeñas piezas individuales de software, muchas de las cuales ya estaban disponibles, como el sistema de edición de textos TeX y el sistema gráfico X Window, que pudieron ser adaptados y reutilizados; otros en cambio tuvieron que ser reescritos.

Para asegurar que el software GNU permaneciera libre para que todos los usuarios pudieran "ejecutarlo, copiarlo, modificarlo y distribuirlo", el proyecto debía ser liberado bajo una licencia diseñada para garantizar esos derechos al tiempo que evitase restricciones posteriores de los mismos. La idea se conoce como copyleft, y está contenida en la Licencia General Pública de GNU (GPL).

En 1985, Richard Stallman creó la Free Software Foundation (FSF o Fundación para el Software Libre) para proveer soportes logísticos, legales y financieros al proyecto GNU. La FSF también contrató programadores para contribuir a GNU, aunque una porción sustancial del desarrollo fue (y continúa siendo) producida por voluntarios. A medida que GNU ganaba renombre, negocios interesados comenzaron a contribuir al desarrollo o comercialización de productos GNU y el correspondiente soporte técnico.

En 1990, el sistema GNU ya tenía un editor de texto llamado Emacs, un exitoso compilador (GCC), y la mayor parte de las bibliotecas y utilidades que componen un sistema operativo UNIX típico. Pero faltaba un componente clave llamado núcleo (kernel). En el manifiesto GNU, Stallman mencionó que "un núcleo inicial existe, pero se necesitan muchos otros programas para emular Unix". Él se refería a TRIX, que es un núcleo de llamadas remotas a procedimientos, desarrollado por el MIT y cuyos autores decidieron que fuera libremente distribuido; Trix era totalmente compatible con UNIX versión 7. En Diciembre de 1986 ya se había trabajado para modificar este núcleo. Sin embargo, los programadores decidieron que no era inicialmente utilizable, debido a que solamente funcionaba en "algunos equipos sumamente complicados y caros" razón por la cual debería ser portado a otras arquitecturas antes de que se pudiera utilizar. Finalmente, en 1988, se decidió utilizar como base el núdeo Mach desarrollado en la CMU (Universidad Carnegie Mellon). Inicialmente, el núcleo recibió el nombre de Alix (así se llamaba una novia de Stallman), pero por decisión del programador Michael Bushnell fue renombrado a Hurd. Desafortunadamente, debido a razones técnicas y conflictos personales entre los programadores originales, el desarrollo de Hurd acabó estancándose.

Fuentes:

http://es.wikipedia.org/wiki/Gnu
http://es.wikipedia.org/wiki/Richard_Stallman
http://www.fsf.org/
http://es.wikipedia.org/wiki/Free_Software_Foundation
http://es.wikipedia.org/wiki/Emacs

http://es.wikipedia.org/wiki/GCC

Programas y Software de GNU

La expresión programas GNU es equivalente a software de GNU. El software de GNU es el software liberado bajo el auspicio del proyecto GNU. A un programa que sea software de GNU, también se lo denomina programa GNU o paquete GNU. El manual del paquete GNU debería indicar que lo es.

La mayoría del software de GNU está protegido por copyleft, pero no todo; sin embargo, todo el software GNU debe ser software libre. Parte del software GNU lo escribe personal de la Fundación para el Software Libre (FSF), pero la mayoría del software lo aportan voluntarios. Del software aportado por voluntarios, a veces el titular de los derechos de autor es la FSF y en otras son los propios colaboradores que lo escribieron.

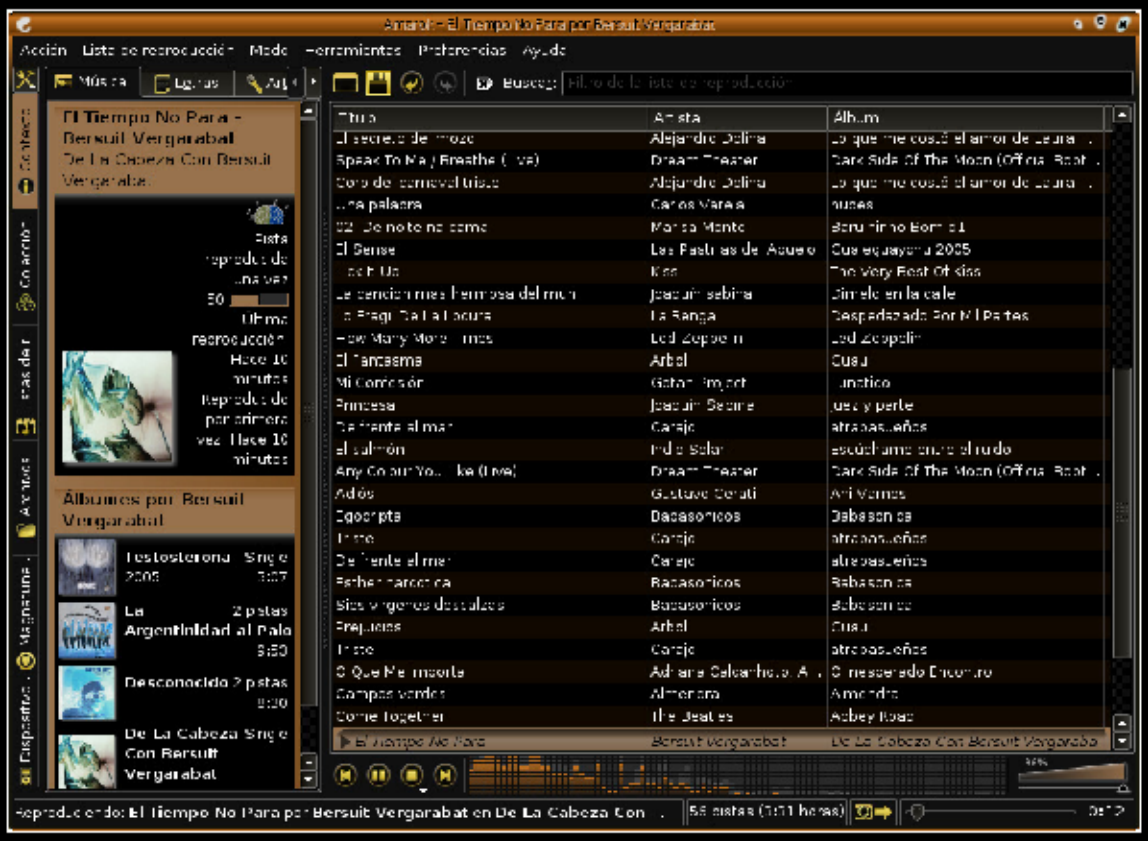
Existe un directorio que sirve para identificar software GNU, el mismo puedo consultarse en el siguiente enlace: http://directory.fsf.org/GNU/.

También existe un enlace en donde nos explica detalladamente como podemos colaborar con el proyecto GNU: http://www.gnu.org/help/help.es.html

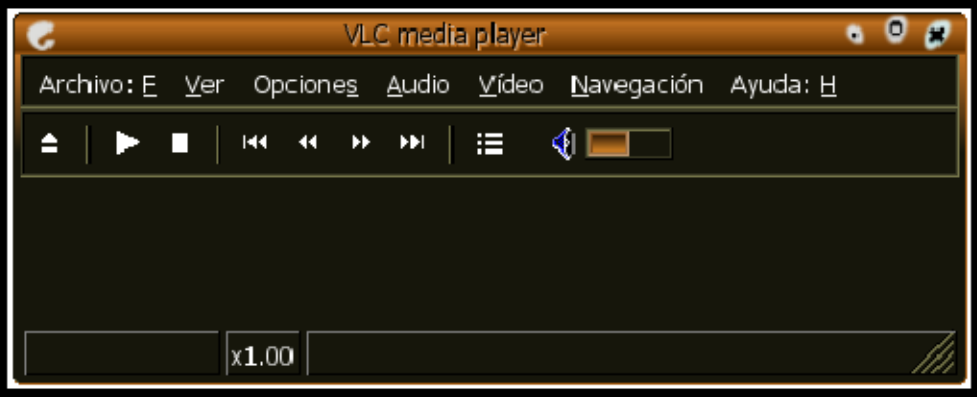
A continuación se ve algunos programas GNU que pueden ser de uso cotidiano. Si quiere un listado detallado, se lo puede consultar en el siguiente enlace: http://directory.fsf.org/ ó bien, puede consultar el siguiente enlace: http://alts.homelinux.net/, que es una pagina que se dedica a ofrecer alternativas de software libre, en comparativa, contra software comercial/privativo.

XMMS (Audio y Video) http://www.xmms.org/

Amarok (Audio y Video) http://amarok.kde.org/



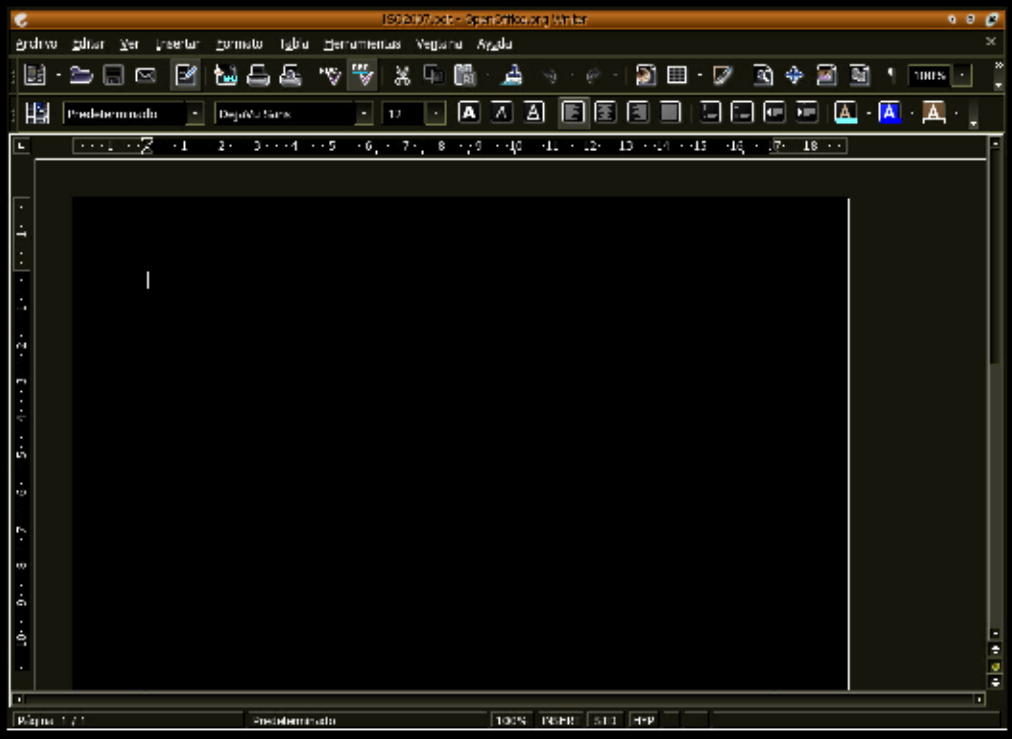
VLC (Reproductor multimedia) http://www.videolan.org/vlc/



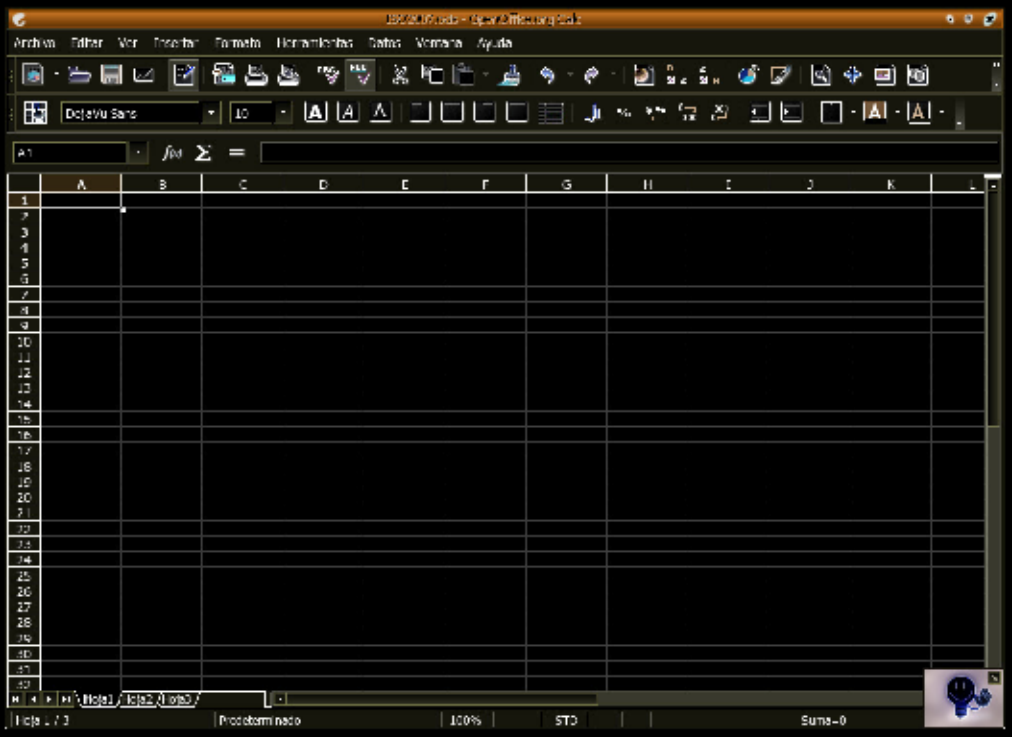
Tótem (Reproductor de películas) http://www.gnome.org/projects/totem/

K3B (Grabador de CD/DVD) http://www.k3b.org/

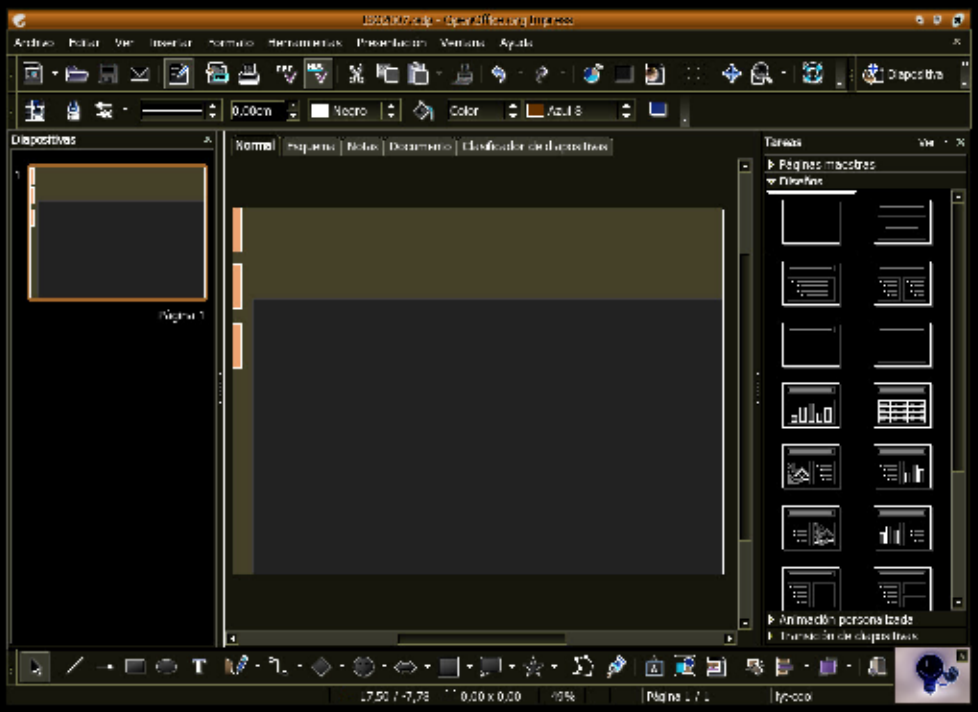
OpenOfficeWriter (Editor de Texto) http://es.openoffice.org/



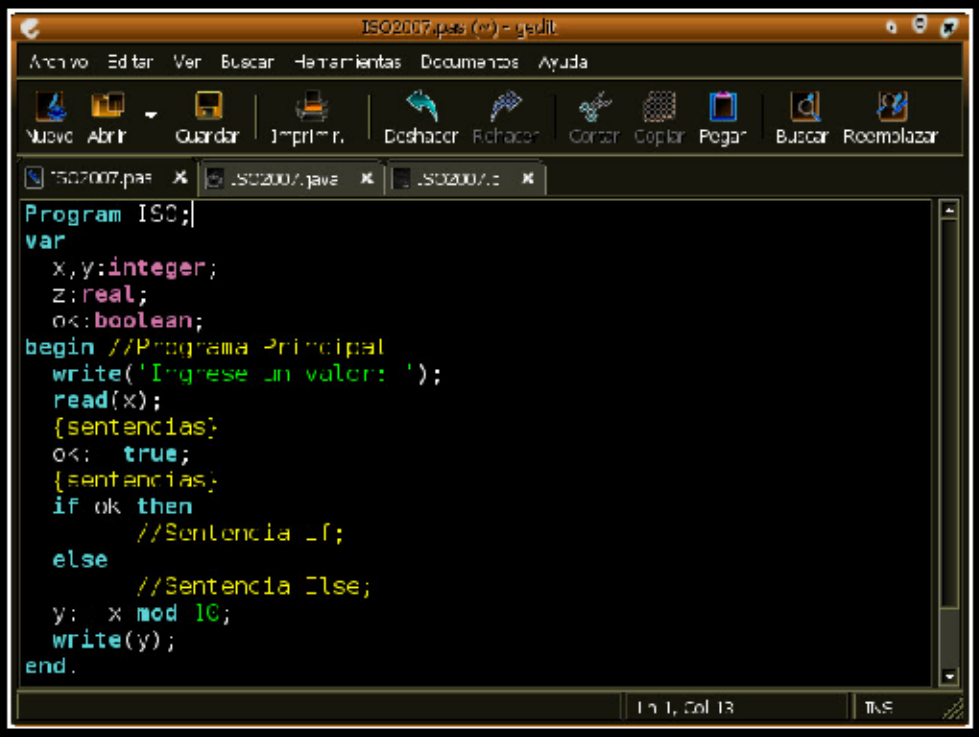
OpenOfficeCalc (Planilla de Calculo) http://es.openoffice.org/



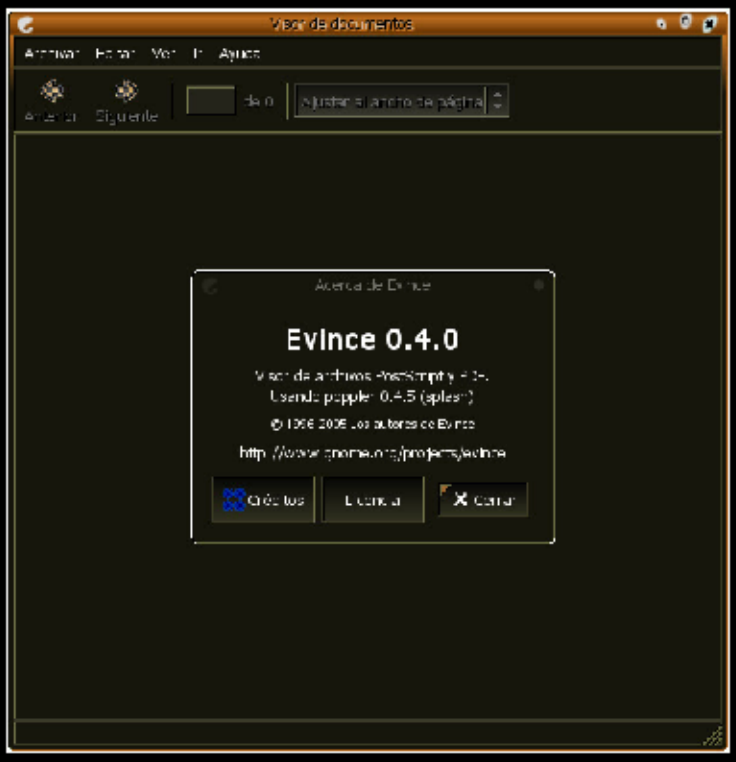
OpenOfficeImpress (Presentaciones para charlas, reuniones, etc.) http://es.openoffice.org/



Gedit (Editor de texto) http://www.gnome.org/projects/gedit/



Evince (Visor de documentos para múltiples formatos) http://www.gnome.org/projects/evince/



Gaim/Pidgin (Mensajero): http://www.pidgin.im/

Iceweasel (Navegador Web) http://www.gnu.org/software/gnuzilla/

Epiphany (Navegador Web) http://www.gnome.org/projects/epiphany/

3.-Conceptos de Linux

Introducción a Linux

Es importante desambiguar el termino Linux, de GNU/Linux. Actualmente se suele hablar de Linux como un Sistema Operativo (SO), pero el termino correcto para el SO, es GNU/Linux. Linux es en si un núcleo (kernel); es el kernel, que junto al sistema GNU, creado por Stallman, dieron origen a al SO GNU/Linux. En el siguiente enlace se puede leer un poco más de las controversias entre el uso del término Linux y GNU/Linux:
http://es.wikipedia.org/wiki/Controversia_por_la_denominaci%C3%B3n_GNU/Linux#Casos_donde_.22GNU.2FLinux.22_es_inaplicable

En abril de 1991, Linus Torvalds de 21 años, empezó a trabajar en unas simples ideas para un SO. Comenzó con un intento por obtener un SO libre similar a Unix que funcionara con microprocesadores Intel 80386. Ese mismo año empezó a escribir el núcleo Linux y decidió distribuirlo bajo la licencia GPL. Rápidamente, múltiples programadores se unieron a Linus en el desarrollo, colaborando a través de Internet y consiguiendo paulatinamente que Linux llegase a ser un núcleo compatible con UNIX. En 1992, el núcleo Linux fue combinado con el sistema GNU, resultando en un SO libre y completamente funcional. El SO formado por esta combinación es usualmente conocido como "GNU/Linux" o como una "distribución Linux" y existen diversas variantes.

También es frecuente hallar componentes de GNU instalados en un sistema UNIX no libre, en lugar de los programas originales para UNIX. Esto se debe a que muchos de los programas escritos por el proyecto GNU han demostrado ser de mayor calidad que sus versiones equivalentes de UNIX. A menudo, estos

componentes se conocen colectivamente como "herramientas GNU". Muchos de los programas GNU han sido también portados a otras plataformas como Microsoft Windows y Mac OS X.

Cronología de las versiones de Linux

En septiembre de 1991, se lanzó la versión 0.01 de Linux. Tenía 10.239 líneas de código. En octubre de ese año, se lanzó la versión 0.02 de Linux, luego en diciembre se lanzó la versión 0.11. Esta versión fue la primera en ser self-hosted (un servicio en su propio proceso). Linux 0.11 podía ser compilado por una computadora que corriera Linux 0.11. Cuando lanzó la siguiente versión, Torvalds adoptó GNU GPL como su propio boceto de licencia, el cual no permitía su redistribución comercial.

Rápidamente el X Window System fue implementado en Linux y en marzo de 1992 la versión 0.95 de Linux fue la primera en ser capaz de correr X. Este gran salto de versiones (de 0.1x a 0.9x) fue por la sensación de que una versión 1.0 sin piezas faltantes no estaba lejos. Sin embargo, esto fue un poco optimista: desde 1993 a principios de 1994, se desarrollaron 15 versiones diferentes de 0.99.

El 14 de marzo de 1994, se lanzó Linux 1.0.0 con 176.250 líneas de código. En marzo de 1995 se lanzó Linux 1.2.0 con 310.950 líneas de código. La versión 2 de Linux se lanzó el 9 de junio de 1996 y fue un gran éxito.

A este le siguieron grandes desarrollos:

- 25 de enero de 1999: se lanzó Linux 2.2.0 con 1.800.847 líneas de código.
- 18 de diciembre de 1999: se publicaron parches de IBM Mainframe para 2.2.13, permitiendo de esta forma que Linux fuera usado en máquinas corporativas.
- 4 de enero de 2001: se lanzó Linux 2.4.0 con 3.377.902 líneas de código.
- 17 de diciembre de 2003: se lanzó Linux 2.6.0 con 5.929.913 líneas de código. A la fecha la última versión estable del kernel Linux es la 2.6.22.3.

Fuentes:

http://www.linux.org/
http://es.wikipedia.org/wiki/Proyecto_GNU
http://es.wikipedia.org/wiki/Linux

4-Introducción breve de GNU-Linux

Introducción

GNU/Linux, es el termino que se debe emplear cuando queremos hacer referencia el SO GNU con el kernel Linux. Como ya hemos dicho, decir Linux para hacer referencia el SO no es correcto, ya que el SO en si mismo es GNU; ejemplo de esto es que en lugar de tener un SO GNU con un kernel Linux, podríamos tener un SO GNU con un kernel Hurd: http://es.wikipedia.org/wiki/GNU_Hurd.

Breve reseña historia

Los primeros sistemas GNU/Linux se originaron en 1992, al combinar utilidades de sistema y librerías del proyecto GNU con el núcleo Linux. Desde fines de 1990 Linux ha obtenido el apoyo de diversas empresas multinacionales del mundo de la informática, tales como IBM, Sun Microsystems, Hewlett-Packard y Novell. Actualmente GNU/Linux es comercializado en computadores de escritorio y portátiles. Si bien GNU/Linux es usado como sistema operativo en computadores de escritorio (PCs x86 y x86-64 así como Macintosh y PowerPC), computadores de bolsillo, teléfonos celulares, dispositivos empotrados y otros, su mayor desarrollo se ha llevado a cabo en el mundo de los servidores y supercomputadores. Si queremos leer un poco mas sobre la historia de GNU/Linux y Linux, podemos ir al siguiente enlace: http://es.wikipedia.org/wiki/Historia_de_Linux

Distribuciones

Una distribución Linux o distribución GNU/Linux (abreviada con frecuencia distro) es un conjunto de aplicaciones reunidas que permiten brindar mejoras para instalar fácilmente un sistema operativo basado en GNU/Linux. Son "sabores" de GNU/Linux que, en general, se diferencian entre sí por las herramientas para configuración y sistemas de administración de paquetes de software para instalar. La elección de una distribución depende de las necesidades del usuario y de gustos personales.

Existen numerosas distribuciones Linux. Cada una de ellas puede incluir cualquier cantidad de software adicional (libre o no), como algunos que facilitan la instalación del sistema y una enorme variedad de aplicaciones, entre ellos, entornos gráficos, suites ofimáticas, servidores web, servidores de correo, servidores FTP, etcétera.

La base de cada distribución incluye el núcleo Linux, con las bibliotecas y herramientas del proyecto GNU y de muchos otros proyectos/grupos de software, como BSD. Usualmente se utiliza la plataforma XFree86 o la X.Org para sostener interfaces gráficas. Existe una infinidad de distribuciones, cada una, tiene una característica propio. E inclusive, algunas distros, son armadas en base a otras anteriores. Ejemplo de esto, es que se puede categorizar como "distribuciones principales" a:

Debian: http://www.debian.org/
Gentoo: http://www.gentoo.org/
Red Hat Linux: http://www.redhat.com/
Slackware: http://www.slackware.com/
SuSe (openSuSe): http://en.opensuse.org/Welcome_to_openSUSE.org
Mandriva (unión entre Mandrake y Conectiva): http://www.mandriva.com/es/community/

Luego, de estas "distros principales" se generaron otras que también tienen gran relevancia en la actualidad, y sirvieron como base para nuevas distros, ellas son:

Ubuntu (basada en Debian): http://www.ubuntu.com/
Knoppix (basada en Debian): http://www.knoppix.org/
Fedora (continuación del proyecto RedHat): http://fedoraproject.org/
Red Hat Enterprise (basada en RedHat): http://www.redhat.com/rhel/

Y de ellas podemos nombrar:

- Basadas en Debian:

  GnuLinEx: http://www.linex.org/
  Linspire: http://www.linspire.com/

- Basadas en Ubuntu:

  Edubuntu: http://www.edubuntu.org/
  Guadalinex (las primeras versiones se basaban en GnuLinEx): http://www.guadalinex.org/
  Kubuntu: http://www.kubuntu.org/

- Basadas en Knoppix:

  Damn Small Linux: http://www.damnsmalllinux.org/
  Morphix: http://www.morphix.org/

- Basadas en Gentoo:

  Kororaa: http://kororaa.org/
  Ututo: https://www.ututo.org/www/

- Basadas en Slackware:

  SLAX: http://www.slax.org/
  VectorLinux: http://www.vectorlinux.com/

En los siguientes enlaces se puede consultar sobre más distribuciones oficiales de GNU/Linux:

http://www.linux.org/dist/
http://distrowatch.com/

Estructura

Cuando hablamos de un SO GNU hacemos referencia a tres elementos fundamentales:

- El kernel (núcleo)
- El Shell (interprete de comandos)
- El FileSystem (sistema de archivos)

El **kernel** (también conocido como núcleo) es la parte fundamental de un sistema operativo. El kernel o núcleo de linux se podría definir como el corazón de este sistema operativo. Es, a grandes rasgos, el encargado de que el software y el hardware de una computadora puedan trabajar juntos.

El **Shell** (intérprete de comandos) es el programa que recibe lo que se escribe en la terminal y lo convierte en instrucciones para el sistema operativo. Un intérprete de comandos es un programa que lee las entradas del usuario y las traduce a instrucciones que el sistema es capaz de entender y utilizar.

El **Filesystem** se traduce como "sistema de archivos"; y es la forma en que dentro de un SO se organizan y se administran los archivos.

En capítulos posteriores se habla con mas detenimiento de cada uno de estos componentes de un SO GNU/Linux.

Fuente:

http://es.wikipedia.org/wiki/Distribuci%C3%B3n_Linux

5.-Kernel (Linux)

En capítulos anteriores, estuvimos hablando sobre el kernel, y particularmente se comento de Linux como un kernel en si mismo. A continuación vas a ver algunas características sobre dicho kernel de Linux.

Arquitectura

Hoy por hoy, Linux es un núcleo monolítico híbrido. Los controladores de dispositivos y las extensiones del núcleo normalmente se ejecutan en un espado privilegiado, con acceso irrestricto al hardware, aunque algunos se ejecutan en espacio de usuario.

A diferencia de los núcleos monolíticos tradicionales, los controladores de dispositivos y las extensiones al sistema operativo se pueden cargar y descargar fácilmente como módulos, mientras el sistema continúa funcionando sin interrupciones. También, a diferencia de los núcleos monolíticos tradicionales, los controladores pueden ser prevolcados (detenidos momentáneamente por actividades más importantes) bajo ciertas condiciones. Esta habilidad fue agregada para manejar correctamente interrupciones de hardware, y para mejorar el soporte de Multiprocesamiento Simétrico.

A diferencia de los núcleos monolíticos tradicionales, los controladores de dispositivos son fácilmente configurables como Loadable Kernel Modules, y se pueden cargar o descargar mientras se está corriendo el sistema.

Lenguajes de Programación

Linux está escrito con una versión del lenguaje de programación C apoyado por GCC, junto a unas pequeñas secciones de código escritas con el lenguaje ensamblador (assembly language). Se usan muchos otros lenguajes en alguna forma, básicamente en la conexión con el proceso de construcción del kernel. Estos incluyen a Perl, Python y varios lenguajes shell scripting. Algunos drivers también pueden ser escritos en C++, Fortran, u otros lenguajes, pero esto es altamente desaconsejable. El sistema de construcción de Linux oficialmente solo soporta GCC como kernel y compilador de driver.

Portabilidad

Si bien, Linux no se ideó originalmente como un sistema Portable, ha evolucionado en esa dirección. Linux es ahora de hecho, uno de los núcleos de sistema operativo más ampliamente portados, y funciona en sistemas muy diversos que van desde iPAQ (una handheld) hasta un zSeries (un mainframe masivo, muy costoso).

Arquitectura de máquina virtual

El kernel de Linux puede correr sobre muchas arquitecturas de máquina virtual tanto como host del sistema operativo como cliente. La máquina virtual usualmente emula la familia de procesadores Intel x86, aunque en algunos casos también son emulados procesadores de PowerPC o AMD.

Versiones

Más allá de haber desarrollado su propio código y de integrar los cambios realizados por otros programas, Linus Torvalds continua lanzando nuevas versiones del núcleo de Linux. Estos son llamados núcleos "vanilla", lo que significa que no han sido modificados por nadie. Muchos vendedores del sistema operativo Linux modifican el kernel de sus productos, principalmente para agregarle soporte a dispositivos o herramientas que no fueron oficialmente lanzadas como estables, mientras que algunas distribuciones, como Slackware, mantienen el kernel vanilla.

Numeración

La versión del kernel de Linux actualmente consta de cuatro números. Por ejemplo, asumamos que el número de la versión está compuesta de esta forma: A.B.C[.D] (ej.: 2.2.1, 2.4.13 ó 2.6.12.3).

- El número A denota la versión del kernel. Es el que cambia con menor frecuencia y solo lo hace cuando se produce un gran cambio en el código o en el concepto del kernel. Históricamente solo ha sido modificado dos veces: en 1994 (versión 1.0) y en 1996 (versión 2.0).

- El número B denota la mayor revisión del kernel.

Antes de la serie de Linux 2.6.x, los números pares indicaban la versión "estable" lanzada. Por ejemplo una para uso de fabricación, como el 1.2, 2.4 ó 2.6. Los números impares, en cambio, como la serie 2.5.x, son versiones de desarrollo, es decir que no son consideradas de producción.

Comenzando con la serie Linux 2.6.x, no hay gran diferencia entre los números pares o impares con respecto a las nuevas herramientas desarrolladas en la misma serie del kernel. Linus Torvalds dictaminó que este será el modelo en el futuro.

- El número C indica una revisión menor en el kernel. En la forma anterior de versiones con tres números, esto fue cambiado cuando se implementaron en el kernel los parches de seguridad, bugfixes, nuevas características o drivers. Con la nueva política, solo es cambiado cuando se introducen nuevos drivers o características; cambios menores se reflejan en el número D.

- El número D se produjo cuando un grave error, que requiere de un arreglo inmediato, se encontró en el código NFS de la versión 2.6.8. Sin embargo, no habían otros cambios como para lanzar una nueva revisión (la cual hubiera sido 2.6.9). Entonces se lanzó la versión 2.6.8.1, con el error arreglado como único cambio. Con 2.6.11, esto fue adoptado como la nueva política de versiones. Bug-fixes y parches de seguridad son actualmente manejados por el cuarto número dejando los cambios mayores para el número C.

También, algunas veces luego de las versiones puede haber algunas letras como "rc1" o "mm2". El "rc" se refiere a release candidate e indica un lanzamiento no oficial. Otras letras usualmente (pero no siempre) hacen referencia a las iniciales de la persona. Esto indica una bifurcación en el desarrollo del kernel realizado por esa persona, por ejemplo ck se refiere a Con Kolivas, ac a Alan Cox, mientras que mm se refiere a Andrew Morton.

El modelo de desarrollo para Linux 2.6 fue un cambio significativo desde el modelo de desarrollo de Linux 2.5. Previamente existía una rama estable (2.4) donde se habían producido cambios menores y seguros, y una rama inestable (2.5) donde estaban permitidos cambios mayores. Esto significó que los usuarios siempre tenían una versión 2.4 testeada y con lo último en seguridad y bug fixes, aunque tuvieran que esperar por las características de la rama 2.5. La rama 2.5 fue eventualmente declarada estable y renombrada como 2.6. Pero en vez de abrir una rama 2.7 inestable, los desarrolladores de kernels eligieron continuar agregando los cambios en la rama "estable" 2.6. De esta forma no había que seguir manteniendo una rama vieja pero estable y se podía hacer que las nuevas características estuvieran rápidamente disponibles y se pudieran realizar más test con el último código.

Sin embargo, el modelo de desarrollo del nuevo 2.6 también significó que no había una rama estable para aquellos que esperaban seguridad y bug fixes sin necesitar las últimas características. Los arreglos solo estaban en la última versión, así que si un usuario quería una versión con todos los bug fixed conocidos también tendría las últimas características, las cuales no habían sido bien testeadas. Una solución parcial para esto fue la versión ya mencionada de cuatro números (y en 2.6.x.y), la cual significaba lanzamientos puntuales creados por el equipo estable (Greg Kroah-Hartman, Chris Wright, y quizás otros). El equipo estable solo lanzaba updates para el kernel más reciente, sin embargo esto no solucionó el problema del faltante de una serie estable de kernel. Los vendors de la distribución de Linux, como Red Hat y Debian, mantienen los kernels que salen con sus lanzamientos, de forma que una solución para algunas personas es seguir el kernel de una distribución.

Como respuesta a la falta de un kernel estable y de gente que coordinara la colección de bugfixes, en diciembre de 2005 Adrian Bunk anunció que continuaría lanzando kernels 2.6.16.y cuando el equipo estable lanzara 2.6.17. Además pensó en incluir updates de drivers, haciendo que el mantenimiento de la serie 2.6.16 sea muy parecido a las viejas reglas de mantenimiento para las serie estables como 2.4.

La última versión estable del kernel es la 2.6.26.2

Términos de la Licencia

Inicialmente Linux se distribuyó bajo los términos de una licencia que prohibía la explotación comercial. Pero esta licencia fue reemplazada, poco tiempo después, por la GNU GPL. Los términos de esta última licencia permiten la distribución y venta de copias o incluso modificaciones, pero requiere que todas las copias del trabajo original y trabajos de autoría derivados del original sean publicados bajo los mismos términos, y que código fuente siempre pueda obtenerse por el mismo medio que el programa licenciado.

Fuente:

http://es.wikipedia.org/wiki/Kernel

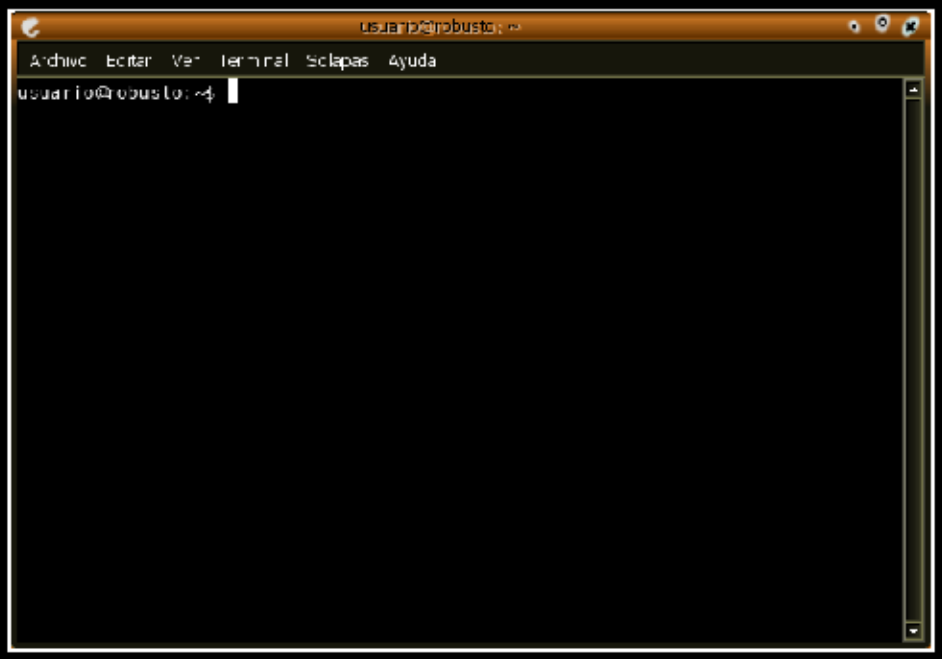
6-Shell

El shell (-también conocido como el intérprete de comandos línea de comandos, terminal, consola-) es un programa que actúa como interfaz, para comunicar al usuario con el sistema operativo mediante una ventana que espera comandos textuales ingresados por el usuario en el tedado, los interpreta y los entrega al SO para su ejecución. La respuesta del SO es mostrada al usuario en la misma ventana. A continuación, la shell queda esperando más instrucciones. Se interactúa con la información de la manera más simple posible, sin gráficas, solo el texto.

Es posible que un sistema operativo tenga varios intérpretes de comandos; dentro de GNU/Linux y Unix, existen tres grandes familias de Shells, estas son: Korn-Shell (ksh), Bourne-Shell (sh) y C-Shell (csh). Estas se diferencian entre sí básicamente en la sintaxis de sus comandos y en la interacción con el usuario. El mas usado hoy en dia es bash (-su nombre es un acrónimo de **b**ourne-**a**gain **sh**ell, haciendo un juego de palabras sobre el Bourne-shell-).

La shell no forma parte del kernel básico del SO; sino que la misma "dialoga" con el kernel. La shell es iniciada por un proceso denominado "login", y dado que cada usuario tiene asignado una shell por defecto, la misma se inicia cada vez que un usuario comienza a trabajar en su estación de trabajo (es decir se "loguea" en una terminal). Dentro del contenido del archivo /etc/passwd, se puede ver cual es la shell que cada usuario tiene asignada por defecto.

El funcionamiento del shell consiste en que, en su forma mas básica, se muestra un prompt (-conjunto de caracteres que se muestran en una línea de comandos para indicarnos que está a la espera de ordenes. En el Bourne Shell y sus derivados, el prompt suele ser el carácter $ para los usuarios y # para el administrador-), en donde el usuario teclea una orden en el teclado y finaliza la orden (-normalmente con la tecla Intro/Enter-), y la computadora ejecuta la orden, proporcionando una salida de texto.



Fuentes:

http://es.wikipedia.org/wiki/L%C3%ADnea_de_comandos
http://es.wikipedia.org/wiki/Bash
http://es.wikipedia.org/wiki/Bourne_Shell
http://es.wikipedia.org/wiki/Korn_Shell
http://es.wikipedia.org/wiki/C_Shell

7-Filesystem

Filesystem se traduce como "sistema de archivos". Es la forma en que dentro de un sistema de cómputo se organizan, se administran los archivos. Esa administración comprende:

- Métodos de acceso: cómo se acceden los datos contenidos en el archivo.
- Manejo de archivos: cómo actúan los mecanismos para almacenar, referenciar, compartir y proteger los archivos.
- Manejo de la memoria secundaria: Cómo se administra el espacio para los archivos en memoria secundaria.
- Mecanismos de integridad: con qué métodos se garantiza la incorruptibilidad del archivo.

La mayoría de los sistemas operativos poseen su propio sistema de archivos. Los sistemas de archivos estructuran la información que luego será representada ya sea textual o gráficamente utilizando un gestor de archivos (-por ejemplo Midnight Commander, Nautilus, Konqueror, etc.-).

Los filesystem más comunes utilizan dispositivos de almacenamiento de datos que permiten el acceso a los datos como una cadena de bloques de un mismo tamaño, a veces llamados sectores (-usualmente de 512 bytes de longitud-). El software del sistema de archivos es responsable de la organización de estos sectores en archivos y directorios y mantiene un registro de qué sectores pertenecen a qué archivos y cuáles no han sido utilizados.

Generalmente un sistema de archivos (filesystem) tiene directorios que asocian nombres de archivos con archivos, usualmente conectando el nombre de archivo a un índice en una tabla de asignación de archivos de algún tipo, como los inodos de los sistemas Unix. La estructura de directorios puede ser plana o jerárquica (ramificada o "en árbol").

En algunos sistemas de archivos los nombres de archivos son estructurados, con sintaxis especiales para extensiones de archivos y números de versión. En otros, los nombres de archivos son simplemente cadenas de texto y los metadatos de cada archivo son alojados separadamente.

En sistemas de archivos jerárquicos, en lo usual, se declara la ubicación precisa de un archivo con una cadena de texto llamada "ruta" (path). La nomenclatura para rutas varía ligeramente de sistema en sistema, pero mantienen por lo general una misma estructura.

Una ruta viene dada por una sucesión de nombres de directorios y subdirectorios, ordenados jerárquicamente de izquierda a derecha y separados por algún carácter especial que suele ser una barra ('/') o barra invertida ('\') y puede terminar en el nombre de un archivo presente en la última rama de directorios especificada.

Los sistemas de archivos tradicionales proveen métodos para crear, mover y eliminar tanto archivos como directorios, pero carecen de métodos para crear, por ejemplo, enlaces adicionales a un directorio o archivo (enlace duro en Unix - los enlaces duros son una referencia, ó indicador a los datos físicos sobre un sistema de archivos-) ó renombrar enlaces padres (el ".." en Unix).

A continuación veremos una lista con algunos filesystem utilizados hoy en día:

- **ext2** (second extended filesystem o "segundo sistema de archivos extendido") es un sistema de archivos para el sistema operativo GNU/Linux. La principal desventaja de ext2 es que no implementa el registro por diario o bitácora (Journaling) que sí implementa su sucesor ext3, el cual es totalmente compatible. ext2 fue el sistema de archivos por defecto de las distribuciones Red Hat Linux, Fedora Core y Debian hasta ser reemplazado recientemente por su sucesor ext3. Para mas información: http://es.wikipedia.org/wiki/Ext2

- **ext3** (third extended filesystem o "tercer sistema de archivos extendido") es un sistema de archivos con registro por diario. Es el sistema de archivo más usado en distribuciones GNU/Linux. La principal diferencia con ext2 es el registro por diario. Un sistema de archivos ext3 puede ser montado y usado como un sistema de archivos ext2. Otra diferencia importante es que ext3 utiliza un árbol binario balanceado (árbol AVL) e incorpora el asignador de bloques de disco Orlov. Para más información: http://es.wikipedia.org/wiki/Ext3

- **ReiserFS** es un sistema de archivos de propósito general. Actualmente es soportado por Linux y existen planes de futuro para incluirlo en otros sistemas operativos. A partir de la versión 2.4.1 del núcleo de Linux, ReiserFS se convirtió en el primer sistema de ficheros con journal en ser incluido en el núcleo estándar. También es el sistema de archivos por defecto en varias distribuciones, como SuSE (excepto en openSuSE 10.2 que su formato por defecto es ext3), Xandros, Linspire y Knoppix (entre otros). http://es.wikipedia.org/wiki/ReiserFS

- **XFS** es un sistema de archivos de 64 bits con journaling de alto rendimiento. XFS se incorporó a Linux a partir de la versión 2.4.25. Los programas de instalación de las distribuciones de SuSE, Gentoo, Mandriva, Slackware, Fedora Core, Ubuntu y Debian ofrecen XFS como un sistema de archivos más. En FreeBSD el soporte para solo-lectura de XFS se añadió a partir de Diciembre de 2005 y en Junio de 2006 un soporte experimental de escritura fue incorporado a FreeBSD-7.0-CURRENT. Para más información: http://es.wikipedia.org/wiki/XFS

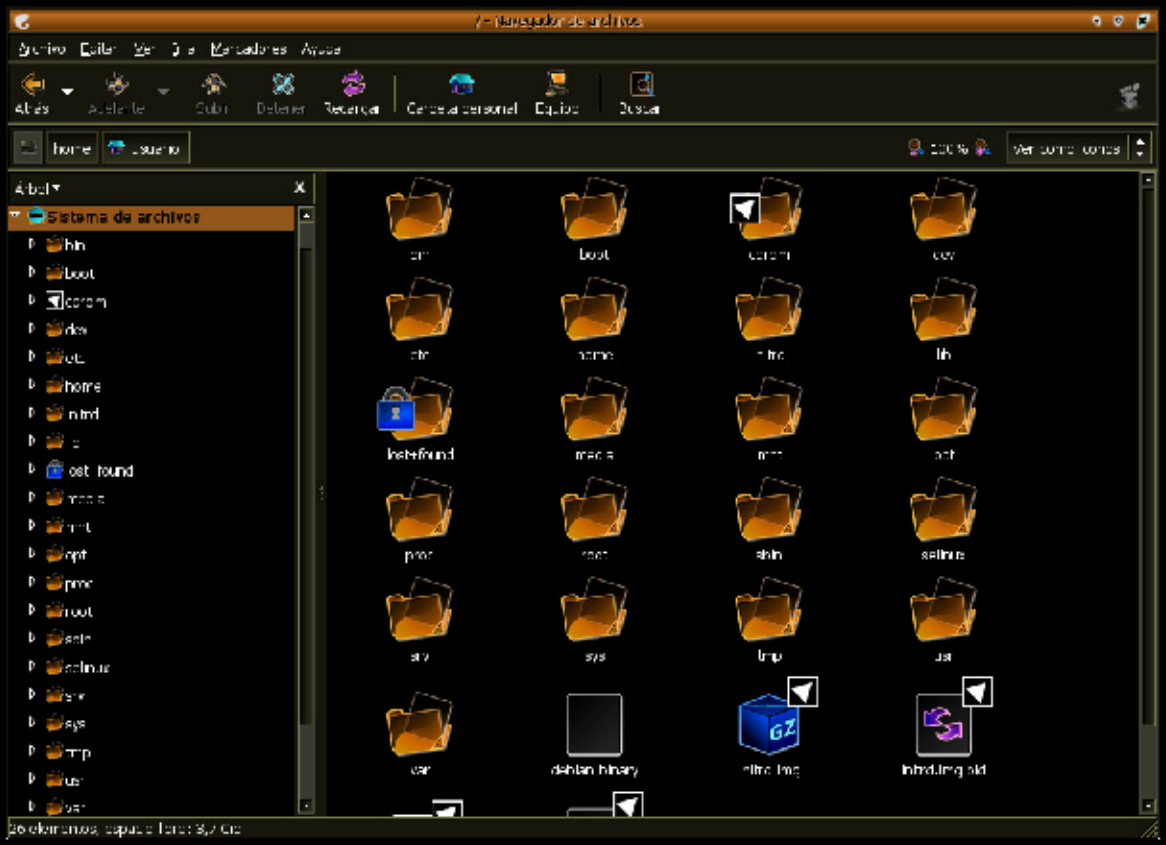
En el momento de instalación GNU/Linux crea una estructura de directorios básica, definida por la Filesystem Hierarchy Standard Group llamada Filesystem Hierarchy Standard (FHS).

FHS define los directorios principales y sus contenidos en el sistema operativo LinuxGNU/Linux y otros sistemas de la familia Unix. Se diseñó originalmente en 1994 para estandarizar el sistema de archivos de las distribuciones GNU/Linux, basándose en la tradicional organización de directorios de los sistemas Unix.

En 1995 se amplió el ámbito del estándar a cualquier Unix que se adhiriese voluntariamente. Todos los archivos y directorios aparecen bajo el directorio raíz /, aunque se encuentre en distintos dispositivos físicos.

Ejemplos de directorios definidos por FHS incluyen:

Bin	Comandos binarios esenciales
Boot	Archivos estáticos del cargador de arranque
Dev	Archivos de dispositivos
Etc	Configuración del sistema especifica
Lib	Librerías esenciales compartidas y módulos del Kernel
Media	Punto de montaje para dispositivos removibles
Mnt	Punto de montaje para montar fliesystems temporarios
Opt	Paquetes de sofware de aplicación agregados
Sbin	programas binarios esenciales del sistema
Srv	Datos para servicios provistos por el sistema
Tmp	Archivos temporarios
Usr	Estructura secundaria
Var	Datos variables
Home	Directorio base de usuarios
Root	directorio home del root
Proc	usado en lugar del/dev/kmen – soporta los procesos



A continuación vamos a ver un poco más en detalle los directorios que se muestran en el siguiente árbol:

```
/___bin
|___dev
|___etc
|    |___X11
|    |___httpd
|    |___mail
|    |___ppp
|    |___sysconfig
|___sbin
|___home
|        |___ISO2007
|        |___usuario
|        |___alumno
|___lib
|___proc
|___tmp
|___usr
|    |___bin
|    |___local
|    |___X11R6
|    |___etc
|    |___include
|    |___lib
|    |___man
|    |___src
|___var
|    |___spool
|    |___log
```

/bin: bin es la abreviación de binaries, o ejecutables. Es donde residen la mayoría de los programas esenciales del sistema, como cp, ls y mv. Cuando se usa la orden cp, se está ejecutando el programa /bin/cp. Si ejecutamos el comandos ls -F se verá que la mayoría de los ficheros de /bin tienen un asterisco añadido al final de sus nombres; esto indica que son archivos ejecutables.

/dev: Los archivos en /dev son conocidos como controladores de dispositivo (device drivers) son usados para acceder a los dispositivos del sistema y recursos, como discos duros, modems, memoria, etc; Por ejemplo, los archivos que comienzan su nombre con fd son controladores de disqueteras. fd0: es la primera disquetera, fd1 la segunda.

/etc: contiene una serie de archivos de configuración del sistema. Estos incluyen /etc/passwd (la base de datos de usuarios), /etc/rc (scripts de inicialización del sistema), etc.

/sbin: se usa para almacenar programas esenciales del sistema, que usará el administrador del sistema.

/home: contiene los directorios "home" de los usuarios. Por ejemplo, /home/ISO2007 es el directorio del usuario ISO2007.

/lib: contiene las imágenes de las librerías compartidas. Estos archivos contienen código que compartirán muchos programas. En lugar de que cada programa contenga una copia propia de las rutinas compartidas, estas son guardadas en un lugar común, en /lib. Esto hace que los programas ejecutables sean menores y reduce el espacio usado en disco.

/proc: es un "sistema de ficheros virtual". Los ficheros que contiene realmente residen en memoria, no en un disco. Hacen referencia a varios procesos que corren en el sistema, y le permiten obtener información acerca de que programas y procesos están corriendo en un momento dado.

/tmp: Muchos programas tienen la necesidad de generar cierta información temporal y guardarla en un fichero temporal. El lugar habitual para esos ficheros es en /tmp.

/usr: es un directorio muy importante. Contienen una serie de subdirectorios que contienen a su vez algunos de los más importantes y útiles programas y archivos de configuración usados en el sistema.

Los directorios descritos arriba son esenciales para que el sistema esté operativo, pero la mayoría de las cosas que se encuentran en /usr son opcionales para el sistema. De cualquier forma, son estas cosas opcionales las que hacen que el sistema sea útil e interesante. Sin /usr, tendría un sistema aburrido, solo con programas como cp y ls. /usr contiene la mayoría de los paquetes grandes de programas y sus ficheros de configuración:

> /usr/X11R6: contiene el sistema XWindow si se ha instalado. El sistema X Window es un entorno gráfico grande y potente el cual proporciona un gran número de utilidades y programas gráficos, mostrados en "ventanas" en su pantalla. Este directorio contiene todos los ejecutables de XWindow, archivos de configuración y de soporte.
>
> /usr/bin: Contiene la mayoría de los programas que no se encuentran en otras partes como /bin.
>
> /usr/etc: Como /etc contiene diferentes archivos de configuración y programas del sistema, /usr/etc contiene incluso más que el anterior. En general, los archivos que se encuentran aquí no son esenciales para el sistema, a diferencia de los que se encuentran en /etc, que si lo son.
>
> /usr/include: contiene los archivos de cabecera para el compilador de C. Estos archivos (la mayoría de los cuales terminan en .h, de "header") declaran estructuras de datos, subrutinas y constantes usados en la escritura de programas en C. Los archivos que se encuentran en /usr/include/sys son generalmente usados en la programación de en Unix a nivel de sistema. Si se está familiarizado con el lenguaje de programación C, aquí encontrarán los ficheros de cabecera, como por ejemplo stdio.h, el cual declara funciones como printf().
>
> /usr/lib: contiene las librerías equivalentes "stub" y "static" a los ficheros encontrados en /lib. Al compilar un programa, este es "enlazado" con las librerías que se encuentran en aquí, las cuales dirigen al programa a buscar en /lib cuando necesita el código de la librería. Además, varios programas guardan archivos de configuración en /usr/lib.

/usr/local: es muy parecido a /usr contiene programas y archivos no esenciales para el sistema. En general, los programas que se encuentran en /usr/local son específicos de su sistema esto es, el directorio /usr/local difiere bastante entre sistemas Unix. Aquí encontrará programas como TeX y Emacs, si se los instala.

/usr/man: Este directorio contiene las páginas de manual. Hay dos subdirectorios para cada página "sección" de las páginas. Por ejemplo, /usr/man/man1 contiene los fuentes (es decir, los originales por formatear) de las páginas de manual de la sección 1, y /usr/man/cat1 las páginas ya formateadas de la sección 1.

/usr/src: contiene el código fuente (programas por compilar) de varios programas de su sistema. El más importante es /usr/src/linux, el cual contiene el código fuente del Núcleo de Linux.

/var: contiene directorios que a menudo cambian su tamaño o tienden a crecer. Muchos de estos directorios solían residir en /usr, pero desde que estamos tratando de dejarlo relativamente inalterable, los directorios que cambian a menudo han sido llevados a /var. Algunos de estos directorios son:

/var/log: contiene varios archivos de interés para el administrador del sistema, específicamente históricos del sistema, los cuales recogen errores o problemas con el sistema. Otros archivos guardan las sesiones de presentación en el sistema, así como los intentos fallidos.

/var/spool: contiene archivos van a ser pasados a otro programa. Por ejemplo, si su máquina está conectada a una red, el correo entrante será almacenado en /var/spool/mail hasta que sea leído o se lo elimine.

Fuentes:

http://es.wikipedia.org/wiki/Orlov_%28asignador_de_bloques_de_disco%29
http://es.wikipedia.org/wiki/Sistema_de_archivos
http://es.wikipedia.org/wiki/Gestor_de_archivos
http://es.wikipedia.org/wiki/Konqueror
http://es.wikipedia.org/wiki/Nautilus_%28inform%C3%A1tica%29
http://es.wikipedia.org/wiki/Midnight_Commander
http://es.wikipedia.org/wiki/Enlace_duro
http://es.wikipedia.org/wiki/Filesystem_Hierarchy_Standard
http://www.ant.org.ar/cursos/curso_intro/filesystem.html

8.-Comandos

A continuación se mostraran algunos comandos de los sistemas Unix y GNU/Linux, agrupados según su funcionamiento. Sólo se comentara el funcionamiento básico del comando. Si se quiere ver más información respecto a los mismos, se puede consultar la siguiente dirección: http://www.ss64.com/bash/index.html

Información del sistema

Varios:

man comando: muestra información sobre el comando.
help comando: muestra información sobre un comando interno del shell(bash).
apropos palabra_clave: busca dentro de las descripciones de man la palabra clave.

cal: muestra el calendario.
uptime: muestra la hora,tiempo de funcionamiento,nº usuarios conectados y la carga media.
date: muestra la fecha y hora del sistema,en formato local.
hwclock –show: muestra el reloj harware (también llamado reloj de la bios y reloj cmos).
atch –n tiempo comando: ejecuta un comando cada x segundos (2 por defecto).

clear: limpia la pantalla.
reset: restaura la consola. útil para cuando empiezan a aparecer caracteres raros.

hostname: visualiza el nombre de la máquina.
tty: muestra el nombre de fichero de la terminal conectada a la salida estándar.
/etc/init.d/servicio [stop|start|restart]: [para|inicia|reinicia] un servicio/demonio
startx: arranca el entorno gráfico.
./script: ejecuta un script de shell.
exit: termina la ejecución del programa en curso.
init 0: apaga la máquina.
init 6: reinicia la máquina.

shutdown –t1 –h now: apaga la máquina .
shutdown –t1 –r now: reinicia la máquina.

su: entrar a la sesión como root u otro usuario.
su nombre_usuario: estando como root entramos como otro usuario.

passwd: cambio de contraseña de un usuario logueado.
who: muestra información de los usuarios conectados al sistema.
users: muestra información de los usuarios conectados al sistema.
id: muestra información del usuario actual,(grupos a los que pertenece,uid,gid)
groups: muestra los grupos a los que pertenece un usuario.

adduser usuario: crea un nuevo usuario.
adduser usuario grupo: añade un usuario existente a un grupo ya existente.
adduser –no-create-home usuario: crea un usuario pero sin directorio personal(home).
addgroup grupo: crea un nuevo grupo.

deluser usuario: elimina un usuario.
deluser usuario grupo: elimina un usuario del grupo especificado.
deluser –remove-home usuario: elimina un usuario y su directorio home.

delgroup grupo: elimina un grupo.
delgroup grupo –only-if-empty: elimina un grupo sólo si no tiene ningún usuario.

usermod –l nuevo_login: cambia el nombre del usuario.
usermod –d nueva_home –m login: cambia el home al usuario, creándolo si no existiese y transfiriendo el contenido.
usermod –e 2008-01-10 login: fecha en que la cuenta del usuario será desactivada, expresada en formato aaaa-mm-dd
groupmod –n nuevo_nombre grupo: cambia el nombre de un grupo.

Memoria y procesos:

memtest86+: Realiza una comprobación del estado de la memoria si optamos a ello desde los gestores de arranque, para lo cual necesitamos editarlos añadiéndoles su ruta, suponiendo que /boot esté en la partición raiz (/)

En lilo.conf (si usamos LILO):

```
image  = /boot/memtest86+.bin
label = memtest86+
```

En menu.lst (si usamos GRUB):

```
title  memtest86+
root   (hd0,0)
kernel /boot/memtest86+.bin
```

make-memtest86+-boot-floppy: Crea un disquete de arranque de memtest86+
free: Muestra el uso de memoria.
ps aux: Muestra información de los procesos en curso.
top: Muestra información de los procesos en curso. (tecla z colorea los activos)
pstree: Muestra los procesos en curso en árbol.
killall proceso: Para un proceso.

Disco duro:

du: Muestra espacio ocupado en disco, del directorio en curso si no indicamos nada.
df: Muestra información sobre particiones montadas.
mount: Vemos el listado de dispositivos montados.
mount -t ext3 /dev/hda2 /disco: Monta la partición ubicada en /dev/hda2, del tipo ext3 en el directorio /disco
umount /dev/hda2: Desmonta un dispositivo

Trabajo con ficheros

Varios:

ls: Lista los ficheros de un directorio concreto.
ls -l: Lista también las propiedades y atributos.
ls -la: Lista ficheros incluidos los ocultos de sistema.
ls -la | more: Lista los ficheros de un directorio de forma paginada.

cat -n fichero: Muestra el contenido de un fichero.(-n lo numera)
more fichero: Muestra el contenido de un fichero de forma paginada.
echo texto: nos muestra en pantalla, el texto que le siga
grep 'cadena' archivo: Muestra las líneas del archivo que contienen la cadena.
file fichero: Muestra de qué tipo es un fichero.

tail archivo: Muestra las últimas líneas de un archivo,10 por defecto.
head -nº fichero: Muestra las primeras (número) líneas de un fichero.
type comando: Muestra la ubicación del comando indicado.
find directorio -name unNombre: Busca todos los ficheros con nombre unNombre en directorio.
pwd: Visualiza el directorio actual.

cd nombre_directorio: Cambia de directorio
cd ..: Vuelves al anterior.
cp -dpR fichero1 ruta_fichero2: Realiza una copia del fichero1 a ruta_fichero2,cambiándole el nombre.
cp -dpR fichero1 /directorio: Copia fichero1 a directorio,conservando fichero1 el nombre.

mv ruta_fichero1 ruta_fichero2: Mueve y/o renombra ficheros o directorios.
mkdir nom_directorio: Crea un directorio.
rmdir nom_directorio: Elimina un directorio(tiene que estar vacío).

rm archivo: Elimina archivos .
rm -r directorio: Borra los ficheros de un directorio recursivamente.
rm *.pepe: Borra todos los ficheros .pepe del directorio actual.

diff [opciones] fichero1 fichero2: Compara ficheros.
diff -w fichero1 fichero2: Descarta espacio en blanco cuando compara líneas.
diff -q fichero1 fichero2: Informa sólo de si los ficheros difieren,no de los detalles de las diferencias.
diff -y fichero1 fichero2: Muestra la salida a dos columnas.

wc fichero: Muestra el nº de palabras, líneas y caracteres de un archivo.
wc -c fichero: Muestra el tamaño en bytes de un fichero.

touch [-am][-t] fichero: Cambia las fechas de acceso (-a) y/o modificación (-m) de un archivo.

chown [-R] usuario fichero: Cambia el propietario de un fichero o directorio.
chgrp [-R] grupo fichero: Cambia el grupo de un fichero o directorio.
chmod [-R][ugo][+/- rwxs] fichero: Cambia los permisos de acceso de un fichero

Cómo afectan los permisos a los directorios:

- r permite ver su contenido(no el de sus ficheros),
- w permite añadir o eliminar ficheros (no modificarlos),
- x permite acceder al directorio.

Método para de determinar los permisos: chmod 760 archivo

	dueño	grupo	otros
ASCII:	r w x	r w -	- - -
Binario:	1 1 1	1 1 0	0 0 0
Octal:	7	6	0

Paso de ASCII a Binario: r w x r w - - - - | activar = 1
1 1 1 1 1 0 0 0 0 | desactivar = 0

Paso de Binario a Octal: 1 1 1 1 1 0 0 0 0 | r activado = 4 |
4+2+1 4+2+0 0+0+0 | w activado = 2 |
7 6 0 | x activado = 1

Compresión:

Comprimir zip: zip -r archivo.zip fichero; ejemplo: zip -r iso.zip ./iso/
Descomprimir zip: unzip archivo.zip
Ver contenido zip: unzip -v archivo.zip

Comprimir gz: gzip -r archivo; ejemplo: gzip -r ./iso
Descomprimir gz: gzip -d archivo.gz
Ver contenido gz: gzip -c archivo.gz

Mientras que zip comprime y empaqueta, gzip ó bzip2 sólo comprimen archivos, no directorios, para eso existe tar:

Empaquetar: tar -vcf archivo.tar /fichero1 /fichero2
Desempaquetar: tar -vxf archivo.tar
Ver contenido: tar -vtf archivo.tar

Para comprimir varios archivos y empaquetarlos en un sólo archivo hay que combinar el tar y el gzip o el bzip2 de la siguiente manera:

Empaquetar y comprimir: tar -zvcf archivo.tgz directorio
Desempaquetar y descomprimir: tar -zvxf archivo.tgz
Ver contenido: tar -zvtf archivo.tgz

Empaquetar y comprimir: tar -jvcf archivo.tbz2 directorio
Desempaquetar y descomprimir: tar -jvxf archivo.tbz2
Ver contenido: tar -jvtf archivo.tbz2

Opciones de tar:

- -c: crea un nuevo archivo.
- -f: cuando se usa con la opción -c, usa el nombre del archivo especificado para la creación del archivo tar. Cuando se usa con la opción -x, retira del archivo el fichero especificado.
- -t: muestra la lista de los archivos que se encuentran en el archivo tar
- -v: muestra el proceso de archivo de los ficheros.
- -x: extrae los ficheros de un archivo.
- -z: comprime el fichero tar con gzip.
- -j: comprime el fichero tar con bzip2.

Entrada/Salida

stdin: entrada estandar para datos, el teclado (0)
stdout: salida estandar para los programas, la pantalla (1)
stderr: salida estandar para los mensajes de error,la pantalla (2)

Redirecciones: un redireccionador redirige la salida de un comando a un fichero

<: comando < fichero
>: comando > fichero
>>: comando >> fichero

Tuberías: Una tubería hace que la salida de un programa sea la entrada de otro.

|: comando1 | comando2.

Sobre Kernel, Logs y Hardware:

uname -a: Versión del kernel.
echo módulo >> /etc/modules: Inserta un módulo en el kernel de forma permanente.
alsaconf: Programa interactivo que detecta las tarjetas de audio y carga los módulos adecuados.
last: Lista los usuarios conectados/desconectados (fichero /var/log/wtmp

cat /proc/version: Versión del nucleo y compilador empleado.
cat /proc/modules: Lista los módulos cargados.
cat /proc/meminfo: Información sobre la memoria.
cat /proc/cpuinfo: Información sobre el procesador.
cat /proc/devices: Información sobre dispositivos en uso.

lsmod: Lista los módulos cargados.
lsmod | grep módulo: Ver si está cargado el módulo.

modinfo módulo: Muestra información sobre un módulo.
modprobe módulo: Inserta un módulo en el kernel cargando antes los módulos de los cuales dependa.
modprobe -r módulo: Elimina un módulo del kernel y si procede los que dependen del mismo.
modconf: Programa gráfico para listar, cargar y descargar módulos del kernel.

insmod módulo: Inserta un módulo en el kernel.
rmmod módulo: Elimina un módulo del kernel.

Fuente:

http://www.esdebian.org/staticpages/index.php?page=20050930063719706