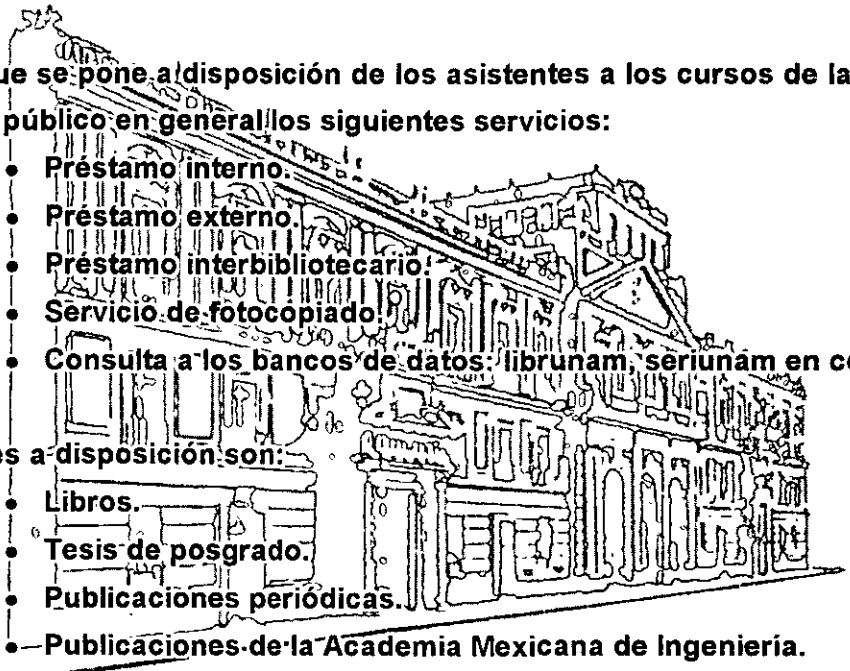
FACULTAD DE INGENIERIA U.N.A.M.
DIVISION DE EDUCACION CONTINUA

CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN
" ING. BRUNO MASCANZONI "

El Centro de Información y Documentación Ing. Bruno Mascanzoni tiene por objetivo satisfacer las necesidades de actualización y proporcionar una adecuada información que permita a los ingenieros, profesores y alumnos estar al tanto del estado actual del conocimiento sobre temas específicos, enfatizando las investigaciones de vanguardia de los campos de la ingeniería, tanto nacionales como extranjeras.



Es por ello que se pone a disposición de los asistentes a los cursos de la DECFI, así como del público en general los siguientes servicios:

- **Préstamo interno.**
- **Préstamo externo.**
- **Préstamo interbibliotecario.**
- **Servicio de fotocopiado.**
- **Consulta a los bancos de datos: librunam, seriunam en cd-rom.**

Los materiales a disposición son:

- **Libros.**
- **Tesis de posgrado.**
- **Publicaciones periódicas.**
- **Publicaciones de la Academia Mexicana de Ingeniería.**
- **Notas de los cursos que se han impartido de 1988 a la fecha.**

En las áreas de ingeniería industrial, civil, electrónica, ciencias de la tierra, computación y, mecánica y eléctrica.

El CID se encuentra ubicado en el mezzanine del Palacio de Minería, lado oriente.

El horario de servicio es de 10:00 a 14:30 y 16:00 a 17:30 de lunes a viernes.

Palacio de Minería Calle de Tacuba 5 Primer piso Deleg. Cuauhtémoc 06000 México, D.F. APDO. Postal M-2285
Teléfonos: 512-8955 512-5121 521-7335 521-1987 Fax 510-0573 521-4020 AL 26

FACULTAD DE INGENIERIA U.N.A.M.
DIVISION DE EDUCACION CONTINUA

A LOS ASISTENTES A LOS CURSOS

Las autoridades de la Facultad de Ingeniería, por conducto del jefe de la División de Educación Continua, otorgan una constancia de asistencia a quienes cumplan con los requisitos establecidos para cada curso.

El control de asistencia se llevará a cabo a través de la persona que le entregó las notas. Las inasistencias serán computadas por las autoridades de la División, con el fin de entregarle constancia solamente a los alumnos que tengan un mínimo de 80% de asistencias.

Pedimos a los asistentes recoger su constancia el día de la clausura. Estas se retendrán por el periodo de un año, pasado este tiempo la DECFI no se hará responsable de este documento.

Se recomienda a los asistentes participar activamente con sus ideas y experiencias, pues los cursos que ofrece la División están planeados para que los profesores expongan una tesis, pero sobre todo, para que coordinen las opiniones de todos los interesados, constituyendo verdaderos seminarios.

Es muy importante que todos los asistentes llenen y entreguen su hoja de inscripción al inicio del curso, información que servirá para integrar un directorio de asistentes, que se entregará oportunamente.

Con el objeto de mejorar los servicios que la División de Educación Continua ofrece, al final del curso deberán entregar la evaluación a través de un cuestionario diseñado para emitir juicios anónimos.

Se recomienda llenar dicha evaluación conforme los profesores impartan sus clases, a efecto de no llenar en la última sesión las evaluaciones y con esto sean más fehacientes sus apreciaciones.

Atentamente

División de Educación Continua.

Palacio de Minería Calle de Tacuba 5 Primer piso Deleg. Cuauhtémoc 06000 México, D.F. APDO Postal M-2285
Teléfonos 5512-8955 5512-5121 5521-7335 5521-1987 Fax 5510-0573 5521-4021 AL 25

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA, UNAM
CURSOS ABIERTOS

CURSO: CC008 INTRODUCCION A UNIX
FECHA: 18 al 22 de marzo del 2002

EVALUACIÓN DEL PERSONAL DOCENTE

(ESCALA DE EVALUACIÓN 1 A 10)

CONFERENCISTA	DOMINIO DEL TEMA	USO DE AYUDAS AUDIOVISUALES	COMUNICACIÓN CON EL ASISTENTE	PUNTUALIDAD
ING. ISABEL G. GALLARDO VALADEZ				

Promedio _____

EVALUACIÓN DE LA ENSEÑANZA

CONCEPTO	CALIF
ORGANIZACIÓN Y DESARROLLO DEL CURSO	
GRADO DE PROFUNDIDAD DEL CURSO	
ACTUALIZACIÓN DEL CURSO	
APLICACIÓN PRACTICA DEL CURSO	

Promedio _____

EVALUACIÓN DEL CURSO

CONCEPTO	CALIF
CUMPLIMIENTO DE LOS OBJETIVOS DEL CURSO	
CONTINUIDAD EN LOS TEMAS	
CALIDAD DEL MATERIAL DIDÁCTICO UTILIZADO	

Promedio _____

Evaluación total del curso_____

Continúa...2

1. ¿Le agradó su estancia en la División de Educación Continua?

SI ☐ NO ☐

Si indica que "NO" diga porqué: _____

_____

2. Medio a través del cual se enteró del curso:

Periódico *La Jornada*	
Folleto anual	
Folleto del curso	
Gaceta UNAM	
Revistas técnicas	
Otro medio (Indique cuál)	

3. ¿Qué cambios sugeriría al curso para mejorarlo?

_____

_____

_____

_____

_____

_____

4 ¿Recomendaría el curso a otra(s) persona(s) ?

SI ☐ NO ☐

5 ¿Qué cursos sugiere que imparta la División de Educación Continua?

_____

_____

_____

_____

_____

_____

6. Otras sugerencias

_____

_____

_____

_____

_____

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE EDUCACIÓN CONTINUA

CURSO
INTRODUCCIÓN A UNIX

COORDINADORA:
ING. ISABEL G. GALLARDO VALADEZ

2002

INTRODUCCIÓN AL SISTEMA OPERATIVO UNIX

TEMARIO

1.- INTRODUCCION A UNIX
- Historia de UNIX
- Características de UNIX
- Ventajas y Desventajas
- Filosofía y potencial
- Plataformas de UNIX

2- COMPONENTES DEL SISTEMA OPERATIVO UNIX
- Núcleo
- Shell
- Sistema de archivos
- Otros componente

3- INICIAR UNA SESIÓN EN UNIX
- Elementos básicos de la cuenta de un usuario
- entrada al sistema
- Conexión remota
- Contraseñas y cambios
- Salida del sistema

4- ATRIBUTOS DE LOS UNIX.
- Importancia de los archivos en UNIX
- Comandos e instrucciones
- Análisis de los permiso de archivos
- Cambio de permisos a los archivos
- Rutas absolutas y relativas
- Edición de archivos
- El editor vi

5- UNIX EN RED

- Características de UNIX en red
- Autenticación en red
- Comandos básicos
- Generalidades

6- OPERADORES DE ENTRADA Y SALIDA

- Redireccionamiento
- Tuberías (Pipes)

7- GENERALIDADES DEL SHELL

- Introducción
- Propósito del Shell
- Tipos de Shell
- Características
- Diferencias entre Shell
- Programas en Shell.

COORDINADORA: ING. ISABEL G. GALLARDO VALADEZ

HISTORIA DE UNIX

La historia del sistema UNIX data de los años sesenta, cuando los Laboratorios Bell de AT&T y el fabricante de computadoras GE (General Electric) trabajaron sobre un sistema operativo experimental denominado MULTICS.

MULTICS, de Multiplexed/Information and Computing System (Información Multiplexada y Sistema de Computación), fue diseñado como sistema operativo interactivo para la computadora GE-645, permitiendo la compartición de información al tiempo que proporcionaba seguridad. El desarrollo sufrió muchos retrasos y las versiones de producción resultaron lentas y con grandes necesidades de memoria. Por una serie de razones, los Laboratorios Bell abandonaron el proyecto. Sin embargo, el sistema MULTICS implementó muchas características innovadoras y produjo un entorno de computación excelente.

En 1969, Ken Thompson, uno de los investigadores de los Laboratorios Bell involucrado en el proyecto MULTICS, escribió un juego para la computadora GE denominado Space Travel. Este juego simulaba el sistema solar y una nave espacial. Thompson vio que el juego se ejecutaba a tirones sobre la máquina GE y resultaba muy costoso, aprox. 7.5 dólares por ejecución. Con la ayuda de Dennis Ritchie, Thompson volvió a escribir el juego para ejecutarse sobre un DEC PDP-7. esta experiencia inicial le dio la oportunidad de escribir un nuevo sistema operativo sobre el PDP-7, utilizando la estructura de un sistema de archivos que habían diseñado Thompson, Ritchie y Rudd Canaday. Thompson, Ritchie y sus colegas crearon un sistema operativo multitarea, incluyendo un sistema de archivos, un intérprete de órdenes y algunas utilidades para la PDP-7. Muchas cosas en el sistema UNIX proceden de este simple sistema operativo.

Puesto que el nuevo sistema operativo multitarea para el PDP-7 podía soportar dos usuarios simultáneamente, se le llamó humorísticamente UNICS de *Uniplexed Information and Computing System (Información Uniplexada y Sistema de Computación)*; el primer uso de este nombre se atribuye a Brian Kernighan. El nombre se cambió ligeramente a UNIX en 1970 y ha permanecido así desde entonces.

En 1972, la segunda edición del *Manual del programador UNIX* mencionaba que había exactamente diez computadoras usando el sistema UNIX, pero indicaba que se estaban esperando más. La popularidad del sistema UNIX creación debido a sus innovaciones ya a que estaba escrito compactamente en un lenguaje de alto nivel con código que podía modificarse de acuerdo a las preferencias individuales. AT&T no ofreció comercialmente el sistema UNIX porque en ese tiempo no estaba en el negocio de las computadoras, sin embargo permitió la disponibilidad del sistema UNIX a universidades, firmas comerciales y al gobierno por un costo simbólico.

Los conceptos del sistema UNIX continuaron creciendo. Los cauces, originalmente sugeridos por Doug McIlory, fueron creados por Ken Thompson al principio de los setenta. La introducción de los cauces hizo posible el desarrollo de la filosofía UNIX, incluyendo el

concepto de una caja de utilidades. Utilizando cauces, las utilidades se pueden conectar, tomando una la entrada de otra utilidad y pasando la salida a una tercera.

Hacia 1974 comenzó a utilizarse ampliamente en los Laboratorios Bell la cuarta edición del sistema UNIX. En 1977, salió la quinta y la sexta ediciones; éstas contenían bastantes más herramientas y utilidades. La séptima edición, el ascendente directo del sistema operativo UNIX disponible hoy día, salió en 1979.

El sistema III de UNIX, basado en la edición séptima, se convirtió en 1982 en la primera versión comercial del sistema UNIX de AT&T. Sin embargo, una vez lanzado el Sistema III, AT&T, a través de su subsidiaria Western Electric, las diferentes ediciones de investigación y las versiones experimentales se distribuyeron a los colegios universitarios y otros laboratorios de investigación.

SISTEMA V DE UNIX

AT&T introdujo el UNIX Sistema V Versión 1 en 1983. Con el UNIX Sistema V Versión 1, AT&T prometió por primera vez mantener compatibilidad ascendente en sus futuras versiones del sistema UNIX. Esto significaba que los programas construidos sobre la Versión 1 podrían continuar operando con futuras versiones del Sistema V

La Versión 1 incorporó algunas características de la versión del sistema UNIX desarrollado en la Universidad de California Berkeley, incluyendo el editor de pantalla **vi** y la biblioteca de manejo de pantalla **curses**. AT&T ofreció el UNIX Sistema V Versión 2 en 1985, ésta versión introdujo protección de archivos contra cortes de potencia, bloqueo de archivos y registros para uso exclusivo por un programa, característica de control de trabajos y administración ampliada del sistema. La versión 2.1 introdujo dos características adicionales de interés para los programadores: la paginación por demanda, que permite ejecutar procesos que requieren más memoria de la que se dispone físicamente y bloqueo de archivos y registros.

En 1987, AT&T introdujo el UNIX Sistema V Versión 3.0; ésta incluía un enfoque de redes simple y consistente. Estas capacidades incluyen STREAMS, utilizado para construir software de redes, el sistema de archivos remoto, utilizado para compartir archivos a través de redes, y la Interfaz a Nivel de Transporte (TLI), utilizada para construir aplicaciones que utilizan redes. La versión 3.1 hizo al sistema V de UNIX adaptable internacionalmente, soportando conjunto de caracteres y formatos de hora y fecha más amplio, aumentos de rendimiento en el uso de memoria y en las copias de seguridad y la recuperación de archivos. La versión 3.2 proporciono seguridad ampliada del sistema, incluyendo la visualización de la hora de última presentación del usuario, el registro de los intentos de inicialización sin éxito y un archivo oculto de contraseñas que impide a los usuarios la lectura de contraseñas cifrada.

La versión 4 unifica varias versiones del sistema UNIX que han sido desarrolladas dentro y fuera de AT&T.

LA DISTRIBUCIÓN SOFTWARE BERKELEY (BSD)

Muchas innovaciones importantes del sistema UNIX se han hecho en la Universidad de California, Berkeley. Algunas de estas ampliaciones han formado parte del Sistema V de UNIX en las primeras versiones y muchas más se han introducido en la Versión 4.

U. C. Berkeley se involucró en el sistema en 1974, comenzado con la cuarta edición. Los estudiantes graduados Bill Joy y Chuck Haley hicieron buena parte del trabajo de la versión de Berkeley, ellos diseñaron un editor denominado **ex** y produjeron un compilador de Pascal, además incluyo el shell C y el editor orientado a pantalla **vi.** En 1978, se hizo la segunda edición conocida como 2BSD. En 1979 se distribuyó 3BSD, proporcionando memoria virtual para permitir que pudiesen ejecutarse grandes programas, esta versión se desarrolló para ejecutarse en el DEC VAX-11/780

A finales de los setenta DARPA (Defense's Advanced Research Projects Agency) decidió basar su entorno de computo en el sistema UNIX, dando el desarrollo de su versión a Berkeley, consolidando el 4BSD. En 1983 apareció la 4.1BSD; disponía de mayor rendimiento. La 4.2BSD también apareció en 1983, introduciendo características de red, incluyendo TCP/IP. La versión 4.3BSD vino en 1987, con pequeños cambios sobre la anterior.

Una de las variantes más importantes es el sistema operativo SunOS de Sun Microsystems.

La versión 4.4BSD es la última que incluye una gran variedad de mejoras, muchas de las cuales se refieren a capacidades de red.

EL SISTEMA XENIX.

En 1960, Microsoft introdujo el sistema XENIX, una variante del sistema UNIX, diseñado para ejecutarse sobre microcomputadoras. La introducción del sistema XENIX llevó las capacidades del sistema UNIX a máquinas de sobremesa; capacidades que anteriormente sólo estaban disponibles en grandes computadoras. El XENIX se basó originalmente en la séptima edición , con algunas utilidades prestadas de la 4.1 BSD. En la versión 3.0 del sistema XENIX, Microsoft incorporó nuevas características del Sistema III del UNIX de AT&T y en 1985 el sistema XENIX se desplazó hacia uno basado en el Sistema V de UNIX.

XENIX se ha portado a diferentes microprocesadores, entre los que se encuentran la familia Intel 8086, 80286 y 80386 y la familia Motorola 68000. En particular, en 1987 XENIx fue portado por la compañía Santa Cruz Operation (SCO) a las máquinas basadas en el 80386, una compañía que ha trabajado con Microsoft sobre el desarrollo de XENIX. En 1987, Microsoft y AT&T comenzaron a desarrollar juntos la fusión de XENIX y el Sistema V de UNIX consumándose en el UNIX Sistema V Versión 3.2. Este esfuerzo logró una versión unificada del sistema UNIX que puede ejecutarse en sistemas que van desde las computadoras personales de sobremesa hasta las supercomputadoras. De todas las

variantes del sistema UUNIX, el sistema XENIX ha conseguido el mayor número de máquinas instaladas. XENIX, aunque todavía soportado por SCO, ya no supone un factor importante en el mundo UNIX.

ESTÁNDARES.

El uso de diferentes versiones del sistema UNIX ocasionaba problemas a las persona encargada de desarrollar aplicaciones para un amplio rango de computadoras que soportaban el sistema UNIX. Para resolver este problema se han desarrollado varios estándares. Estos estándares definen las características que deberá tener un sistema para que puedan construirse aplicaciones que funcionen sobre cualquier sistema que se adecué al estándar. Uno de los objetivos de la versión 4 es unificar las variantes importantes del sistema UNIX es un único producto estándar.

POSIX

POSIX será una familia de estándares que define cualquier aplicación que interactúa con un sistema operativo. Entre las áreas cubiertas por los estándares POSIX están las llamadas al sistema, bibliotecas, herramientas, interfaces, verificación y prueba, características en tiempo real y seguridad. El estándar POSIX que ha recibido mayor atención es el P1003.1 conocido como POSIX.1 que define la interfaz del sistema. Otro estándar POSIX importante es el P1003.2 conocido como POSIX.2 que versa sobre shells y utilidades. El POSIX 1003.3 cubre los métodos de prueba sobre el cumplimiento de los estándares POSIX; el estándar POSIX 1003.4 cubre las extensiones sobre tiempo-real. Existen más de 20 estándares diferentes que han sido desarrollados o están en vías de desarrollo.

X/OPEN.

X/OPEN es un consorcio internacional de vendedores de computadoras establecido en 1984; adopta los estándares e interfaces existentes pero no desarrollas sus propios estándares. Algunas compañía que pertenecen al Consorcio son AT&T, DEC, HP, IBM , Sun, Unisys, Bull, Ericson, ICL, Olivetti y Phillips.

El objetivo de X/OPEN consiste en estandarizar las interfaces software. Realizan esta labor publicando su *Common Applications Environment (CAE).*

En 1993, X/OPEN asume la responsabilidad de controlar la evolución de una especificación común de interfaz de programación de aplicaciones (API). Esta aplicación permitirá a un vendedor de software sobre sistema UNIX desarrollar aplicaciones que trabajen sobre cualquier plataforma con sistema UNIX que cumpla esta especificación. El nombre original de esta especificación fue Spec 1170.

Para que hoy un sistema operativo reciba la calificación de sistema UNIX X/OPEN, sólo necesita ajustarse a la Single UNIX Specification y a los X/OPEN Curses, Especificación 4 Edición, que definen las interfaces de terminal internacionalizados.

LA HISTORIA DE UNIX EN MEXICO

Debido a que es muy distinto el desenvolvimiento de los sistemas en nuestro país, vale la pena comentar como inició UNIX en México.

Básicamente, en los ochentas UNIX comienza a verse en forma en las universidades e institutos en México. Sin embargo se veía como un sistema interesante a nivel científico. Por estudios de firmas de consultoría que atendían al gobierno, se comienza a ver que este sistema puede ser útil en el sector público, por correr en distintas plataformas con la misma curva de aprendizaje. Dentro de las compañías y empresas gubernamentales que con mas ímpetu entran al ambiente de los sistemas abiertos se encuentran Petróleos Mexicanos, Comisión Federal de Electricidad, el IMSS, SHCP, por mencionar algunas. Surgen graves problemas de resistencia al cambio, debido a que gran parte de estas dependencias eran controladas por gigantes trasnacionales con equipos propietarios.

En el inicio, se puede ver que UNIX era soportado por empresas como ALTOS, NCR, SPERRY y algunas empresas que soportaban arquitecturas de ATyT y la plataforma de SCO XENIX. A finales de los ochentas, empresas resultantes de fusiones como UNISYS y Honeywell-Bull o bien trasnacionales importantes como Hewlett-Packard y Digital Equipment Corporation incluyen en su línea de productos y como compromiso especial el fomentar gran parte de su línea de negocios en plataformas de sistemas abiertos, haciendo así que otras grandes empresas de estrategia cerrada se vean forzados a ingresar al mercado de los sistemas abiertos, mas que por ser de su interés, por la pérdida de un mercado que exige día a día mayor facilidad de portabilidad, sencillez de capacitación y bajo costo de aplicaciones producidas por terceros.

Hoy en día México ha ingresado en forma definitiva al mercado de los sistemas abiertos, en principio como estándar en el Gobierno, y por otro lado por las empresas de todos tamaños que encuentran a esta filosofía clave para su expansión y crecimiento armónico. Podemos ver que México cuenta con las mejores firmas representando equipos micro computacionales o bien estaciones de trabajo de las mas conocidas a nivel mundial, como SUN o APOLLO.

Inclusive México cuenta con equipos mini y mainframe en UNIX, lo cual ha sido estratégico para los Bancos de México mas grandes, y los corporativos mas importantes. Mas que una moda, el usuario, como usted, demanda día a día mayor poder computacional a menor costo, detalle clave que se encuentra en UNIX.

DEFINICIÓN DE UNIX

Es un sistema operativo. Un sistema operativo es un conjunto de programas que controlan los recursos de una computadora. Este sistema operativo forma parte de una corriente denominada Sistemas Abiertos, qu en términos generales propone que cualquier programa o aplicación pueda efectuase en cualquier equipo con independencia al proveedor del hardware del equipo.

Esto presenta un panorama muy atractivo, ya que de esta forma el usuario no queda "atado" a un determinado proveedor, razón por la cual puede adquirir ayuda y aplicaciones de terceros, siendo este medio por tanto muy reñido.

¿ POR QUÉ ES IMPORTANTE UNIX?

Durante los últimos veinte años el sistema operativo UNIX se ha convertido en un sistema operativo potente, flexible y versátil. Sirve como sistema operativo par todo tipo de computadoras, incluyendo las computadoras personales monousuario y las estaciones de trabajo de ingeniería, y microcomputadoras multiusuario, minicomputadoras, mainframes y supercomputadoras. El número de computadoras que funcionan con el sistema UNIX ha ido creciendo de forma exponencial, con 10 millones de máquinas con UNIX.

El éxito de UNIX se debe a muchos factores, entre los que se incluyen su portabilidad a un gran abanico de máquinas, su adaptabilidad y simplicidad, el amplio rango de tareas que puede ejecutar su naturaleza multiusuario y multitarea y su adecuación a las redes, que ha ido creciendo en importancia a medida que ha ido prosperando Internet.

CARACTERÍSTICAS

MULTIUSUARIO.

Es la capacidad de una computadora de soportar más de un usuario conectados al sistema simultáneamente y proporcionar .acceso a los mismos archivos o programas para los diferentes usuarios que lo requieran.

MULTITAREAS.

Este sistema puede ejecutar varias tareas simultáneamente por un mismo usuario y lo más importante todas estas tareas ejecutadas al mismo tiempo.

MULTIPROCESO.

Los sistemas UNIX permiten generar diversos procesos simultáneamente, los procesos son iniciados por el shell y se les da un ID de proceso o identificador de proceso, el cual, es utilizado par llevar la cuenta del proceso hasta su terminación.

PORTABILIDAD

El sistema UNIX es mucho más fácil de portar o transportar a nuevas máquinas que otros sistemas operativos, esto es, se necesita menos trabajo para adaptarlo y ejecutarlo sobre una máquina nueva. La portabilidad del sistema UNIX es consecuencia directa de estar escrito casi completamente en un lenguaje de alto nivel, el lenguaje C. La portabilidad a un amplio rango de computadoras hace posible mover las aplicaciones de un sistema a otro.

CODIGO FUENTE ABIERTO.

E l código fuente del sistema UNIX y no el código ejecutable, ha estado disponible a usuarios y programadores. A causa de esto, mucha gente ha sido capaz de adaptar el sistema UNIX de formas diferentes. Este carácter abierto ha conducido a la introducción de un amplio rango de caracteristicas nuevas y de versiones personalizadas que se ajustan a necesidades especiales. A las personas que han desarrollado esta adaptación del sistema UNIX les ha resultado fácil porque el código correspondiente es sencillo, modular y compacto. Esto ha fomentado la evolución del sistema UNIX, haciendo posible la fusión de las capacidades desarrolladas por diferentes variantes del sistema UUNIX necesarias para soportar lo entornos de computación de hoy en un sistema operativo único, el UNIX Sistema V Versión 4.

Nuevas características están siendo constantemente desarrolladas por varias versiones del sistema operativo UNIX, siendo la mayoría de estas características compatibles con UNIX Sistema V Versión 4, conocido comúnmente como "UNIX System 5 Release 4".

SHELL

El shell lee las órdenes y las interpreta como peticiones de ejecución de un programa o programas, lo que realiza posteriormente. Debido a este papel, el shell se denomina ***interprete de órdenes.*** Además de ser un intérprete de órdenes, el shell también es un lenguaje de programación. Como un lenguaje de programación permite controlar cómo y cuándo se llevan a cabo las órdenes. El shell utiliza un lenguaje de programación de alto nivel como lo es el Lenguaje C.

SOPORTE A RED.

El sistema UNIX proporciona un entorno excelente para redes. Ofrece programas y facilidades que proporcionan los servicios necesarios para construir aplicaciones basadas en red, base de la computación distribuida. En las computaciones en red, la información y su procesamiento es compartida por diferentes computadoras de la red. El sistema UNIX ha demostrado ser útil en computación cliente/servidor donde máquinas de una red pueden ser al mismo tiempo clientes y servidores. El sistema UNIX ha sido también el sistema básico para el desarrollo de los servicios Internet y para el crecimiento de Internet. Consecuentemente, con la importancia creciente de la computación distribuida e Internet está creciendo la popularidad del sistema UNIX.

FACILIDAD PARA LA CREACIÓN DE PROGRAMAS.

El sistema UNIX proporciona a los usuarios diferentes herramientas y utilidades que se pueden utilizar para realizar una gran variedad de trabajos. Algunas de estas herramientas son órdenes simples que se pueden utilizar para llevar a cabo tareas específicas. Otras herramientas y utilidades son realmente pequeños lenguajes de programación que se pueden utilizar para construir guiones o programas que resuelven sus propios problemas. Lo más importante es que las herramientas están diseñadas para funcionar juntas, como partes de una máquina o bloques de construcción. No sólo se incluyen en el sistema UNIX muchas herramientas y utilerías sino que mucha otras están disponibles como opciones, entre las que se incluyen todas aquellas que se encuentran disponibles en archivos en Internet sin coste alguno.

SISTEMA DE ARCHIVOS

La unidad básica utilizada para organizar la información en el sistema UNIX, se denomina archivo. El sistema de archivos del sistema UNIX proporciona un método lógico par organizar, almacenar, recuperar, manipular y gestionar la información. Los archivos están organizados en un sistemas de archivos jerárquico, agrupados en directorios. Una característica de simplificación importante del sistema UNIX es la forma general de tratamiento de los archivos.

FILOSOFIA DE UNIX

Conforme ha ido evolucionado, el sistema UNIX ha desarrollado un enfoque característico y consistente que se denomina a veces como *FILOSOFIA* UNIX. Esta filosofía ha influido profundamente sobre la estructura del sistema y la forma de operar. Tener presente esta filosofía ayuda a entender la forma en que el sistema UNIX trata los archivos y programas, los tipos de órdenes y la manera de utilizarlos para llevar a cabo una tarea.

La filosofía UNIX se basa en la idea de que un sistema informático potente y complejo debe ser simple, general y extensible y que esto proporciona importantes beneficios tanto pata los usuarios como para los que desarrollan programas. Otra manera de expresar los objetivos básicos de la filosofía UNIX es resaltar que en toda su complejidad y tamaño el sistema UNIX aún refleja la idea de que "lo pequeño es bello". Este enfoque queda especialmente reflejado en la forma en que el sistema UNIX trata los archivos y en la manera de enfocar sus herramientas.

El sistema UNIX contempla los archivos de una manera extremadamente simple y general dentro de un modelo único. Ve de la misma manera los directorios, los archivos ordinarios, los dispositivos, tales como impresoras y discos, y los teclados y terminales de pantalla. El sistema de archivos oculta al usuario detalles del hardware subyacente; por ejemplo, usted no necesita conocer sobre qué unidad de disco se encuentra un archivo. Esta simplicidad le permite concentrarse sobre lo que realmente le interesa, los datos y la información que contiene el archivo. En una red de área local, el concepto de sistema de archivos remoto le ahorra la necesidad de conocer sobre qué máquina están sus archivos.

El hecho de que la pantalla y el teclado se traten como archivos permite utilizar con ellos los mismos programas u órdenes que con archivos almacenados de manera ordinaria, tomando la entrada desde el terminal o visualizando la información sobre él.

Una característica única del sistema UNIX es la gran colección de órdenes o herramientas de software que proporciona. Esta es otra expresión de la filosofía básica. Estas herramientas son pequeños programas, cada uno diseñado para realizar una función específica, y todos diseñados para operar juntos. En lugar de pocos programas grandes, cada uno tratando de hacer muchas cosas, el sistema UNIX proporciona muchas herramientas simples que pueden combinarse para realizar un amplio rango de cosas. Algunas tareas llevan a cabo un tarea básica y tienen nombre mnemotécnicos. Otras son lenguajes de programación y, por tanto, con sintaxis complicadas.

Un buen ejemplo de la orientación de las herramientas es la orden **sort, sort** es un programa que toma un archivo, lo ordena en función de una de las diferentes reglas posibles y saca el resultado. Se puede utilizar con cualquier archivo de texto. Con frecuencia se utiliza junto a otros programas par ordenar su salida.

Un programa independiente para ordenar significa que otros programas no tengan que incluir las operaciones de ordenación. Esto tiene unos beneficios inmediatos para las personas que desarrollan programas, pero también le ayuda a usted. Utilizando un único programa genérico de ordenación, se evita la necesidad de aprender las diferentes órdenes, opciones y convenciones que deberían ser necesarias si cada programa tuviese que proporcionar su propia ordenación.

El énfasis sobre herramientas modulares es soportado por una de las características típicas del sistema UNIX " *el cauce* ***(pipe)***". Esta característica, de importancia tanto para usuarios como para programadores, es un mecanismo general que permite utilizar la salida de una orden como entrada a otra. Se trata del "pegamento" utilizado para unir herramientas que van a realizar la tarea que se necesita. El sistema UNIX trata la entrada y la salida de una forma simple y consistente, utilizando *entrada estándar y salida estándar* . por ejemplo, la entrada a una orden puede tomarse de un terminal o la salida de otra orden, sin necesidad de utilizar una versión diferente de dicha orden.

PLATAFORMAS DE UNIX

UNIX Sistema V Versión 4 ha obtenido un gran impacto en el mercado de los sistemas UNIX. Casi todos los vendedores principales ofrecen versiones de UNIX que se basan en UNIX Sistema V Versión 4 y que se ajustan a los estándares para sistemas abiertas basados en UNIX Sistema V Versión 4 y todas las versiones de UNIX comparten muchas características con la versión 4. Aunque existen decenas de variantes de UNIX, la mayoría comparten una gran cantidad de características, pudiéndose portar software entre la mayoría de variantes moderas casi directamente.

UNIXWARE.

UnixWare ha sido la denominación utilizada por Novell para sus productos en UNIX Sistema V; estos productos los ofrece ahora Santa Cruz Operation a raíz de la venta de dichos productos por Novell en septiembre de 1995. la última versión de UnixWare es UnixWare 2, basada en la integración de UNIX Sistema V Versión 4.2 y Novell NetWare, soportando la arquitectura cliente/servidor.

UnixWare cumple, entre otros con los estándares POSIX 1003.1 y XPG4 de X/Open. Las versiones futuras cumplirá con las especificaciones Spec 1170 (Single UNIX Specification). Existen dos versiones de UnixWare, UnixWare 2 Personal Edition, diseñado para su uso como sobremesa y cliente, y UnixWare 2 Application Server, diseñado para usar en servidores. UnixWare sólo está disponible en la actualidad en computadoras basadas en procesadores Intel y será portada a computadoras basadas en arquitectura RISC, como las máquinas basadas en procesadores SPARC. Se han escrito más de 3,000 aplicaciones para UnixWare. UnixWare soporta también la interface gráfica MOTIF.

SOLARIS

El sistema operativo original de Sun Microsystems fue denominado SunOS. Estaba basado en UNIX Sistema V Versión 2 y BSD 4.3. En 1991, Sun Microsystems crea SunSoft como empresa subsidiaria par desarrollar y distribuir software, incluyendo sistemas operativos. En su inicio, SunSoft comenzó la tarea de migrar el sistema operativo SunOS a una nueva versión de UNIX basada en UUNIX Sistema V Versión 4. La primera versión de SunSoft de UNIX, Solaris 1.0 fue una versión mejorada de SunSoft, se movió hacia un sistema operativo basado en SVR4. SunSoft tiene una gran base de instalaciones UNIX por encima de millón y medio de máquinas. Actualmente existen versiones de Solaris basados en procesadores Intel, SPARC y en PowerPC.

Solaris también tiene soporte a la interface gráfica MOTIF. A partir de la versión 2.4 de Solaris se cumple con POSIX.1 y POSIX.2. Más de 9,000 aplicaciones se ejecutan sobre plataformas Solaris. En la actualidad se maneja Solaris versión 8.0.

UNIXSCO

Santa Cruz Operation (hoy en dia SCO) basa sus sistemas operativos en UNIX Sistema V/386 versión 3.2, una versión de UNIX Sistema V versión 3 diseñada para usarse sobre procesadores Intel 80386. Los dos entornos operativos principales de SCO son Open Desktop 5.0 y Open Server 5.0. Open Desktop 5.0 está concebido para usarse en computadoras personales basadas en Intel. Open Desktop ejecuta la mayoría de aplicaciones DOS, Windows y XENIX. SCO Open Serverincluye un amplio rango de capacidades de red, entre otros TCP/IP, NFS y soporte para varios protocolos LAN.

SCO tiene una gran base instalada de un millón de sistemas UNIX vendidos.

IRIX

IRIX es la versión propietaria de UNIX Sistema V Versión 4 que proporciona Silicon Graphics para usar sobre estaciones de trabajo basadas en MIPS. IRIX es un sistema operativo a 64-bits, que es una de las características que optimizan su rendimiento para aplicaciones gráficas que requieren un procesamiento intensivo de la UCP. La versión actual de IRIX es IRIX 5.3, donde incorpora lo esencial de UNIX Sistema V Versión 4 y 4.2.

IRIX 5.3 ha sido diseñado para proporcionar funciones adicionales en muchas áreas, incluyendo soporte servidor, lanzamiento de aplicaciones y soporte de medios digitales. IRIX 5.3 se ajusta a XPG3, POSIX, SVDI3 y la mayoría de API SVR4.

HP-UX

La variante del sistema operativo UNIX desarrollado y vendido por Hewlett-Packard par usarse en sus computadoras y estaciones de trabajo se denomina HP-UX, basado en UNIX Sistema V Versión 2.0, con muchas ampliaciones. La versión más reciente de HP-UX es HP-UX 10.0, ha sido diseñado conforma muchos de los estándares para UNIX, y HP planea ofrecer una futura versión que cumpla con Spec 1170. En particular HP-UX se ajusta a XPG4. Cumple con las APIs Nivel 1 Definición 3 de la Interfaz del Sistema V, por lo que se ajusta a la definición de SVR4 y a todas las APIs de redes especificadas por Spec 1170. Ahora, HP-UX incorpora la estructura SVR4 File Directory Layout.

DEC OSF/1

Digital Equipment Corporation (DEC) ha vendido una gran cantidad de computadoras en las que se ejecutan su versión del sistema operativo UNIX, llamado ULTRIX. Con la llegada de sus nuevas computadoras basadas en el procesador Alpha se han concentrado en una nueva variante de UNIX. DEC OSF/1, basada en el sistema operativo OSF/1 desarrollado por Open Systems Foundation. DEC OSF/1 posee extensas ampliaciones sobre OSF/1. En particular, proporciona soporte 64 bits, soporte para tiempo real, gestión de memoria incrementada, multiprocesamiento simétrico y un rápido establecimiento del

sistema de archivos. DEC OSF/1 integra los componentes de OSF/1, el sistema V y BSD, ejecutando bajo un múcleo Mach.

DEC OSF/1 se ajustaa Spec 1170, excepto para soporte de curses y a POSIX 1003.1 y POSIX 1003.2. También se ajusta al estándar X/Open XPG4.

AIX

La versión de IBM del sistema operativo UNIX se denomina AIX, desarrollada inicialmente para utilizarla sobre estaciones de trabajo IBM. AIX Versión 4 es la última versión de AIX; esta versión se ejecuta en procesadores POWER y PowerPC. AIX se basa en UNIX Sistema V Versión 3 y tiene características del BSD 4.3 AIX incluye soporte para MOTIF , la interfaz gráfica de usuario desarrollada por OSF e incluye una implementación parcial del Common Desktop Environment. Existen más de 10,000 aplicaciones diferentes que se ejecutan sobre plataformas AIX.

AIX 4.1 se ajusta al perfil básico del estándar X/OPEN SPG4 y a los estándares Spec 1170 de X/OPEN, POSIX.1 y POSIX.2.

A/UX

A/UX de Apple´s UNIX, es una implementación del sistema operativo UNIX de Apple, A/UX 3 funde la funcionalidad del sistema UNIX con el sistema operativo Macintosh System 7. A/UX 3 se basa en UNIX Sistema V Versión 3.2, pero posee muchas extensiones de la versión 3 y 4 de BSD 4.2. A/UX3 se ajusta a la definición de la interfaz del Sistema V y POSIX. Dado que A/UX 3 incorpora el sistema operativo System 7 para el Macintosh, casi todas las aplicaciones Macintosh pueden usarse bajo A/UX, así como portar aplicaciones UNIX al entorno A/UX. A/UX se ejecuta en computadoras Macintosh que utilizan procesadores Motorola 680x0; no se ejecutan em computadoras de Apple ya que se basan en el PowerPC.

LINUX

La variante Linux del sistema operativo UNIX fue diseñada originalmente para ser ejecutada en computadoras personales basadas en procesadores Intel 80x86. ha sido y sigue siendo portada a varias plataformas. El desarrollo de Linux lo inició Linux Torvalds en 1991. En aquel año era estudiante de la Universidad de Helsinki en Finlandia. Desde entonces un gran número de personas han aportado de forma voluntaria sus esfuerzos en la construcción de Linux. Al contrario de lo que ocurre con otras versiones de UNIX, Linux es de libre uso. No obstante, posee un copyright bajo los términos de Licencia pública general GNU, para prevenir que pueda ser vendido y distribuido sin permitir al comprador que lo copie libremente.

Linux cumple el estándar POSIX.1 y el objetivo de sus desarrolladores es conseguir que cumpla con Spec 1170 de X/Open. Linux comparte muchas características de UNIX Sistema V y tiene muchas ampliaciones. Ha llegado a ser una versión popular muy extendida para uso de computadoras personales.

COMPARACION DE UNIX Y WINDOWS NT

UNIX es un sistema operativo creado a finales de los sesentas. Windows NT comienza su existencia a mediados de los años 90. UNIX ha sufrido variantes diversas, en lo que respecta a la particularidad que diversos proveedores han hecho con el mismo. Se puede ver una sola versión de Windows NT, pero hay varias versiones de UNIX, dependiendo del proveedor de hardware principalmente.

Tanto Windows NT como UNIX comparten la característica de portabilidad, sin embargo, Microsoft ha hecho un esfuerzo especial en mantener la interfase programable para la aplicación totalmente estándar, en tanto que esto varia en algunas versiones de UNIX. Esto ha sido objeto a su vez de críticas a Microsoft, por tratar de "monopolizar" el mercado, en tanto que UNIX permitió que esto fuera variable.

UNIX tiene mayor potencial "central" esto es, varios usuarios haciendo varias tareas diversas en el mismo equipo central. Windows NT distribuye el trabajo entre los abonados de la red, cada estación de trabajo puede trabajar independiente y a su vez apoyarse en el equipo central.

Windows NT corre aplicaciones de Windows 3.1, Windows 95, e inclusive DOS. Esto marca una existencia enorme de aplicaciones comparado con la existencia de aplicaciones para UNIX.

CARACTERISTICAS

SISTEMA MULTIUSUARIO, MULTI-PROGRAMACION, MULTITAREAS

Aunque Unix no es el primer sistema capaz de atender a varios usuarios en forma simultánea en diferentes tareas, y, de la misma forma, puede efectuar disitntos programas concurrentemente (atributo de multiprogramacion). Se puede decir que es el primero pensado en atacar mercados pequeños y medianos, tales como la pequeña y mediana industria, permitiendo al empresario por primera vez crecer en forma modular y con un enfoque de economías de escala atractivo.

SISTEMA "DEMOCRATA"

La filosofía de Unix no pretende, en principio, manejar privilegios a nivel usuario. Esto quiere decir que el sistema atenderá con la misma prioridad o importancia a cualquier usuario. De quererlo, el usuario convencional tiene opción a disminuir la importancia a su trabajo, pero solo bajo ciertas circunstancias el super usuario del sistema podrá "forzarlo" a efectuar sus tareas mas rápidamente, de manera temporal. Esto se debe principalmente a la intención de hacer que el usuario haga un esfuerzo en desarrollar sus aplicaciones en forma óptima, sin desperdiciar recursos, ya que de hacerlo, el sistema por si solo, le resta importancia a su trabajo, con el fin de poder atender en forma adecuada al resto de los usuarios.

COMPARACION DE UNIX Y DOS

Tanto DOS (Sistema Operativo de las microcomputadoras personales) como Unix pertenecen a la tendencia de Sistemas Abiertos. En ambos, aplicaciones de terceros pueden correr sin importar la marca del equipo. Sin embargo DOS no es un sistema multiusuario, multiprogramación ni multitareas, en otras palabras, DOS no puede en una misma computadora atender a varios usuarios distintos en donde cada uno de ellos emplea distintas aplicaciones, e inclusive desde una misma terminal empleando distintas aplicaciones en forma simultánea por el mismo usuario. En el caso de Unix todo esto es posible.

La forma en la que DOS puede hacer algunas de estas características es mediante el empleo de un sistema operativo que permita compartir los recursos de una computadora central conocida generalmente como "servidor" o "server". Sin embargo, una red local no es lo mismo que un equipo multiusuario, esto debido a que en este tipo de redes cada usuario debe de contar con una microcomputadora personal para poder efectuar sus aplicaciones, en el caso de Unix basta con terminales tontas pues toda la carga, acceso a archivos, etcetera, la lleva la computadora que contiene Unix, decrementando así el costo por usuario de equipo.

Universidad Nacional Autónoma de México
Facultad de Ingeniería
División de Educación Continua

INTRODUCCIÓN A UNIX.

INTRODUCCIÓN A UNIX.

COORDINADORA ACADEMICA:

ING. ISABEL G. GALLARDO VALADEZ

INSTRUCTORES:

ING. RENE LEDEZMA
ING. ISABEL G. GALLARDO VALADEZ

TEMARIO

1- INTRODUCCION A UNIX

- Historia de UNIX
- Características de UNIX
- Ventajas y Desventajas
- Filosofía y potencial
- Plataformas de UNIX

2- COMPONENTES DEL SISTEMA OPERATIVO UNIX

- Núcleo
- Shell
- Sistema de archivos
- Otros componente

TEMARIO

3- INICIAR UNA SESIÓN EN UNIX

- Elementos básicos de la cuenta de un usuario
- entrada al sistema
- Conexión remota
- Contraseñas y cambios
- Salida del sistema

4- ATRIBUTOS DE LOS UNIX.

- Importancia de los archivos en UNIX
- Comandos e instrucciones
- Análisis de los permiso de archivos
- Cambio de permisos a los archivos
- Rutas absolutas y relativas
- Edición de archivos
- El editor vi

TEMARIO

- 5- UNIX EN RED
  - Características de UNIX en red
  - Autenticación en red
  - Comandos básicos
  - Generalidades
- 6- OPERADORES DE ENTRADA Y SALIDA
  - Redireccionamiento
  - Tuberías (Pipes)

TEMARIO

- 7- GENERALIDADES DEL SHELL
  - Introducción
  - Propósito del Shell
  - Tipos de Shell
  - Características
  - Diferencias entre Shell
  - Programas en Shell.

QUÉ ES UNIX

En sentido estricto, es el núcleo (kernel) de un sistema operativo de tiempo compatido: un progrma que controla los recurso de una computadora y los asigna entre los usuarios, proporciona un sistema de archivos que administra el almacenamiento de la información.

En sentido más amplio, "UNIX" abarca no sólo el núclo, sino también programas esenciales; entre ellos: compiladores, editoes, lenguajes de comandos programas para copiado e impresión de archivos.

Más ampliamente, "UNIX" puede incluir programas desarrollados por el usuario para ser ejecutados en el sistema, por ejemplo, herramientas para preparar documentos, rutinas para análisis estadísticos y paquetes gráficos entre muchos otros.

NOTAS

BREVARIO HISTORICO

- **Originado en los laboratorios BELL AT&T, antecesor del Sistema Operativo MULTICS a finales de los 60's.**
- **KEN THOMPSON Y DENNIS RITCHIE diseñadores originales construyen un juego de viaje espacial para la PDP-7**
- **Posteriormente crearón una nueva estructura de sistemas de archivos añadiendo entorno de procesos con planificación.**
- **En 1969, se escribió la primera versión de UNIX llamada UNICS (Uniplexed Operating and Computing System)**

NOTAS

BREVARIO HISTORICO

- **UNIX nace como una simplificación de MULTICS en 1970.**
- **La implementación original fue codificada en ensamblador**
- **En 1971, el primer cliente real fue la oficina de Abogados de patente BELL con el programa "TROFF".**
- **En 1973, el kernell fue recodificado en lenguaje "C"**
- **En 1975, el proyecto es pasado a una máquina PDP-11**
- **En 1970 y 1975 el sistema se desarrolla para máquinas superiores PDP-11/45 y PDP-11/70**

NOTAS

BREVARIO HISTORICO

- **Provocando la venta de cientos de máquinas PDP-11**
- **Las PDP-11 junto con UNIX se introducen fuertemente al mercado telefónico**
- **Simultáneamente AT&T distribuye copias a muchas universidades del mercado**
- **Se genera la versión BSD (Berkeley Software Distribution) en la Universidad de California Berkeley**
- **AT&T fortalece a UNIX hacia la computación comercial**
- **BSD domina en comunidades universitarias y técnicas**

NOTAS

BREVARIO HISTORICO

- **Comienza la competencia AT&T y BSD**
- **Finales de los 70's AT&T comienza un nuevo esquema de nominación**
  - **System III, finales de los 80's**
  - **System V**
  - **SVR2 y SVR3**
  - **Solo productos de transición**
- **Finales de los 70's y principios de los 80's fue portado prácticamente a casi todas las máquinas con potencial para soportarlo.**

NOTAS

BREVARIO HISTORICO

- **En 1986 se genera XENIX para equipos basados en el 8088 con participación de Microsoft**
- **En 1989 Santa Cruz Opertion libera su versión SCO UNIXSystem V**
- **En 1993 Univel (USI y NOVELL) liberan UNIXWARE**

NOTAS

TIPOS DE UNIX

•AIX	IBM
•A/UX	Apple
•BSD (Lite, 386, Free)	Universidad de Berkeley, Calf.
•Dynix	Sequent
•HP-UX	Hewlett Packard
•Irix	Silicon Graphics
•Hurd (GNU)	Fee Software Foundation
•Mach	Universidad Carnegie-Mellon
•MKS Toolkit	Mortice Kern Systems
•Nextstep	Next
•Unicos	Cray Research
•Unixware	Novell
•OSF/1, Ultrix	Digital Equipment Corporation
•Interactive UNIX, Solaris, SunOS	Sun Microsystems
•SCO	Santa Cruz Operation
•Linux	Unix para PC´s

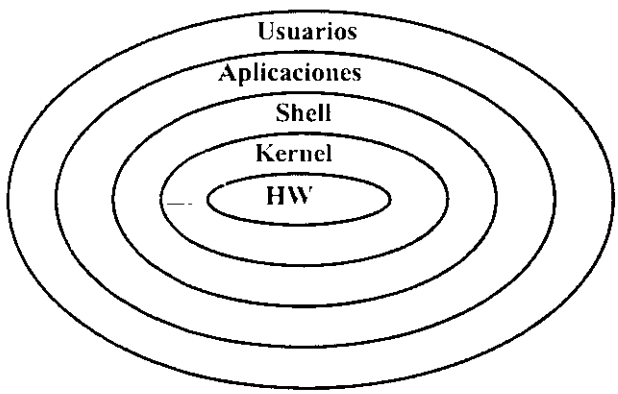
NOTAS

CARACTERÍSTICAS DE UNIX

- **Multiusuario**
- **Multiprocesamiento**
- **Multitareas**
- **Escrito en un lenguaje de alto nivel "C"**
- **Dispone de un elnguaje de control programable llamado "Shell"**
- **Soporte a red como parte del sistema**
- **Ofrece facilidades para la creación de programas y sistemas y un ambiente muy propio para las tareas de diseño de software**
- **Emplea un sistema jerárquico de archivos, con facilidades de protección de archivos, cuentas y procesos**
- **Incluye utilerías y varios lenguajes de programación**

NOTAS

CAPAS DEL SISTEMA OPERATIVO UNIX



NOTAS

CAPAS DEL SISTEMA OPERATIVO UNIX

KERNEL

- **Corazón del Sistema Operativo**
- **Es un programa escrito casi en su totalidad en lenguaje "C" con excepción de una parte del manejo de interrupciones que está escrito en lenguaje ensamblador**
- **Se compone de un pequeño conjunto de programas que hacen posible al sistema proveer otro tipo de servicios**
- **Facilita 4 tipos de servicios básicos:**
  - **Creación y manejo de los procesos**
  - **Sistema de archivos**
  - **Comunicaciones**
  - **La forma de arrancar el sistema**

NOTAS

SHELL

- **Interface al usuario**
- **Interprete de comandos**
- **Permite:**
  - **Manipulación de archivos**
  - **Ejecución de comandos**
  - **Redireccionamiento de entrada/salida**
  - **Control de trabajos**
- **A cada usuario le es asignado un shell por omisión, es inicializado cada vez que el usuario entra en sesión**
- **Provee una serie de comandos especiales para crear los programas llamados scripts**

NOTAS

TIPOS DE SHELL

INTERPRETE (SHELL)	NOMBRE DEL PROGRAMA
sh	Bourne Shell
ksh	Korn Shell
bash	Bash Shell (Extensión del Bourne
zsh	Zshell (Bourne mejorado)
rc	rc (versión recortado del Bourne)
csh	C shell (basado en lenguaje C)
tcsh	tc shell (Cshell mejorado)

NOTAS

UTILERIAS DE UNIX

- **Son programas cuyo principal objetivo es auxiliar en la obtención, procesamiento**
- **Podemos decir que sirven para hacernos la vida más fácil dentro del sistema opertivo UNIX**
- **Existen diversas utilerías, como el ambiente gráfico**

NOTAS

SISTEMA DE ARCHIVOS DE UNIX
FILE SYSTEM

Objetivo: Proporcionar un manejo eficiente de la información

Existen dos puntos de vista:

Sistema: Sistema físico de archivos que tiene una localización tangible en el disco duro

Usuario: Estructura de árbol que forman los directorios en las particiones del disco duro

NOTAS

SISTEMA DE ARCHIVOS DE UNIX
FILE SYSTEM

- **El sistema operativo UNIX esta basado en un modelo arborescente y recursivo**
- **Existe una raíz del sistema de archivos conocida como root ("/") y de ahí se desprenden un conjunto de directorios que contienen archivos y directorios, que a su vez, funciona como la subraíz de un nuevo árbol.**
- **Todo archivo en UNIX está controlado por múltiples niveles de protección que especifican los permisos de acceso del mismo**
- **Hay 3 tipos de componentes:**
  - **Archivos ordinarios: de texto y binarios**
  - **Directorios: donde se agrupan los archivos**
  - **Archivos especiales: asociados a dispositivos que proporcionan el canal de comunicación, los hay de dos tipos: bloques y caracter.**

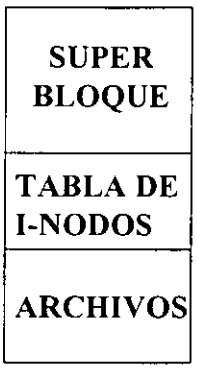
NOTAS

ESTRUCTURA DEL FILE SYSTEM

/root	Es la parte base del sistema de archivos, de donde se desprende el árbol de directorios
/bin	Contiene programas ejecutables y librerías
/etc	Contiene programas y archivos de datos para la administración y configuración del sistema.
/dev	Contiene archivos de configuración de los dispositivos del sistema
/export	Es el directorio que contiene arvhivos y otros sistemas de archivos que comparte el servidor con otras estaciones de trabajo sobre la red.
/lib	Contiene las bibliotecas de programas que utilizan los programadores
/sys	Contiene los llamados archivos fuente del sistema
/tmp	Se usa para almacenamiento temporal
/usr	Es un directorio de propósito general que contiene los subdirectorios de los usuarios
/var	Contiene los registros de mantenimiento del sistema ytambién se puede usar para instalar aplicaciones

NOTAS

VOLUMEN



VOLUMEN. Es el volumen lógico definido que se compone de un conjunto de bloques de tamaño fijo

SUPERBLOQUE. Contiene la información que caracteriza al volumen: tamaño, descripción, etc.

TABLA DE I-NODOS. Contiene la información que describe a los archivos contenidos en el volumen

ARCHIVOS. Secuencia de caracteres

NOTAS

I-NODOS

Es un descriptor único para cada archivo del sistema

El I-nodo de un archivo contiene un conjunto de informaciones que caracterizan a :

- **El propietario**
- **Las protecciones**
- **La relación de los bloques físicos que constituyen el archivo**
- **Las fechas de creación, de última utilización y de la última modificación**
- **El número de enlaces**
- **El tipo de archivo**

NOTAS

PROCESO

Es un programa que se esta ejecutando.
Cada vez que se ejecuta un comando se inicializa un programa

SWAP

Área de memoria utilizada para realizar la paginación y la ejecución de procesos, cuando la memoria principal se ha saturado

NOTAS

QUÉ ES EL SISTEMA UNÍX

Para comprender cómo opera el sistema UNIX, necesita entender su estructura. El sistema operativo UNIX lo forman varios componentes principales. Entre estos componentes están núcleo, el shell, el sistema de archivos (File System) y las órdenes (Programas de usuario). Las relaciones entre el usuario, el shell, el núcleo y el hardware subyacente.

EL NÚCLEO

El **núcleo** es la parte del sistema operativo que interactúa directamente con el hardware de una computadora, a través de los *controladores de dispositivo* (drivers) que están incorporados en el núcleo. Proporciona conjuntos de servicios que pueden ser utilizados por programas, aislando a estos programas del hardware subyacente. Las funciones principales del núcleo son la gestión de la memoria, el control de acceso a la computadora, el mantenimiento del sistema de archivos, el manejo de las interrupciones (señales que finalizan la ejecución), el manejo de errores, la realización de los servicios de entrada y salida (que permiten a las computadoras interactuar con terminales, dispositivos de almacenamiento e impresoras) y la asignación de recursos de la computadora (tales como la UCP o dispositivos de entrada/salida) entre los usuarios.

Los programas interactúan con el núcleo a través de 100 llamadas al sistema, aproximadamente. Las llamadas al sistema dicen al núcleo que lleve a cabo diferentes tareas para el programa, tales como abrir un archivo, escribir en un archivo, obtener información sobre un archivo, ejecutar un programa, terminar un proceso, cambiar la prioridad de un proceso y obtener la hora y la fecha. Las diferentes implementaciones del UNIX Sistema V tienen llamadas al sistema compatibles, teniendo cada una de ellas la misma funcionalidad. Sin embargo, las internas, programas que realizan las funciones de las llamadas al sistema (normalmente) escritas en lenguaje C y la arquitectura del sistema en dos implementaciones diferentes, pueden guardar poca semejanza la una a la otra.

SHELL

El ***shell*** lee las órdenes y las interpreta como peticiones de ejecución de un programa o programas, lo que realiza posteriormente. Debido a este papel, el shell se denomina intérprete de órdenes. Además de ser un intérprete de órdenes, el shell también es un lenguaje de programación. Como lenguaje de programación, permite controlar cómo y cuándo se llevan a cabo las órdenes.

SISTEMA DE ARCHIVOS (FILE SYSTEM)

La unidad básica utilizada para organizar la información en el sistema UNIX se denomina ***archivo***. El sistema de archivos del sistema UNIX proporciona un método lógico para organizar, almacenar, recuperar, manipular y gestionar la información. Los

archivos están organizados en un ***sistema de archivos jerárquico***, agrupados en ***directorios***.

Una característica de simplificación importante del sistema UNIX es la forma general de tratamiento de los archivos. Por ejemplo, los dispositivos físicos se tratan como archivos; esto permite que las mismas órdenes operen sobre archivos ordinarios y sobre dispositivos físicos, es decir, la impresión de un archivo (sobre la impresora) se trata de manera similar a la visualización sobre una pantalla de terminal.

HERRAMIENTAS

El sistema UNIX contiene varios cientos de herramientas o programas de usuario. Las órdenes también se conocen como utilerías o utilidades, pues pueden utilizarse independientemente o en forma conjunta de diversas maneras para llevar a cabo tareas útiles. Estas herramientas se pueden ejecutar invocándolas por su nombre a través del shell, es por ello por lo que se conocer como órdenes.

Una diferencia crítica ente el sistema UNIX y los sistemas operativos anteriores es la facilidad con que los nuevos programas pueden ser instalados; el shell sólo necesita conocer dónde buscar las órdenes y esto es definible por el usuario.

Se pueden realizar muchas tareas utilizando las herramientas estándar suministradas con el sistema UNIX. Hay herramientas para edición y procesamiento de texto, para gestión de información, para comunicaciones electrónicas y para redes, para realizar cálculos, para desarrollar programas de computadoras, para la administración del sistema y para muchos otros propósitos. Las herramientas especializadas, entre las que se incluyen aquellas incluidas en el UNIX y aquellas otras disponibles como opciones, barrían dependiendo de la versión y el fabricante.

APLICACIONES

Se pueden utilizar aplicaciones construidas utilizando órdenes, herramientas y programas del sistema UNIX. Los programas de aplicación llevan a cabo diferentes tipos de tareas. Algunas realizan funciones generales que pueden ser utilizadas por una amplia variedad de usuarios del gobierno, la industria y la educación. Estas se conocen como aplicaciones horizontales e incluyen programas tales como procesadores de texto, compiladores, sistemas de gestión de bases de datos, hojas de cálculo, programas de análisis estadístico y programas de comunicaciones. Otras son especificas de la industria y se conocen como aplicaciones verticales. Ejemplos de ellas serían los paquetes de software utilizados par administrar un hotel, un banco y los terminales operativos de puntos de venta.

Hay varias clases de aplicaciones que han experimentado un crecimiento explosivo en los últimos años. La primera de ellas involucra aplicaciones de red, incluyendo aquellas que permiten a la gente hacer uso de un amplio rango de servicios disponibles en Internet. Destacan entre ellas los exploradores para el World Wide Web, tales como Mosaic y Netscape. Otra clase importante de aplicaciones es aquella que trata con la multimedia.

ENTRADA AL SISTEMA

EL PROCESO DE LOGIN

Este proceso es el que todo usuario efectúa para enlazarse a la máquina Unix desde cualquiera de sus terminales. Para este proceso se deben de conocer lo siguiente:

NOMBRE DE LOGIN

Es el nombre que usted emplea para identificarse como usuario dentro del sistema Unix. Este mismo nombre permite a otros usuarios comunicarse con usted mediante diferentes utilerías.

EL PASSWORD

El password o palabra clave esta asociada a cada usuario dentro de un sistema Unix y previene a otros usuarios ganar acceso a los archivos de otros usuarios. En un principio el administrador del sistema debe de asignar un password a su usuario, hacérselo saber y, posteriormente usted debe de cambiarlo.

El comando que le permite modificar su password es passwd, al momento de entrar a un equipo nuevo lo primero que debe de hacer es cambiar su password mediante este comando. Este comando es autodocumentado, debe de seguir las instrucciones que el mismo indica en todo momento, tecleando primero su password antiguo (de contar con uno inicialmente) después indicar el nuevo password, de una longitud mínima de 6 caracteres de los cuales cuando menos uno es no alfabético y cuando menos dos deben ser alfabéticos, y finalmente re teclear el password nuevo con el fin de asegurar que no se equivocó al teclear la información, esto debido a que en ningún momento verá desplegado en pantalla ni el viejo ni los nuevos passwords.

ACCESO AL SISTEMA

Para tal efecto teclee junto al letrero del login del equipo el nombre del usuario seguido de un return. De contar con un password el sistema se lo preguntará en ese momento. Simplemente tecleelo seguido de un return. Notará que el nombre del usuario es visible en tanto que el password no lo es. De no contar con un password automáticamente accesará el sistema. Al término de este proceso, encontrará el "prompt" del sistema operativo, generalmente un símbolo de pesos.

Ejemplo:

```
login: moises
Password: xxxxxxxx
```

```
Entrada al sistema como usuario moises
En caso de ser necesario se teclea
el password o contraseña. Esta no
aparece en pantalla.
```

JBICACION DEL USUARIO

Una vez accesado el password correcto del usuario al equipo, sucederá la ejecución de un archivo común a todos los usuarios bajo el directorio etc, conocido como el profile. Este archivo indica diversos datos para el ambiente de ese usuario, las rutas en donde buscar comandos, etc. Posteriormente, en el caso particular de cada usuario, existe un segundo archivo, el .profile, que radica en el directorio propio del usuario, este directorio es el que pertenece a dicho usuario. Normalmente el directorio del usuario tiene el mismo nombre que el de entrada al sistema, a nivel login. El usuario puede incluir en el .profile de su propiedad la ejecución de otros comandos, nuevas rutas, y en general cualquier información relevante para situar a dicho usuario en las condiciones particulares de operación que el desee. El usuario sabrá que ha concluido la ejecución de su .profile (explicado a detalle posteriormente) al encontrar el prompt (símbolo que indica la espera de comandos a ser ejecutados por el sistema operativo) en su pantalla. Este prompt normalmente es un símbolo de pesos "$" o en otros casos un símbolo de porcentaje "%", dependiendo del tipo de shell que emplea el usuario y descrito a detalle en otra sección de éste capítulo.

LA RUTA DEL USUARIO

Terminados los sucesos anteriores el usuario tiene que conocer la trayectoria con respecto a la raíz , esta puede ser encontrada mediante la ejecución del comando pwd. Este comando indica la ruta donde se encuentra ese usuario en ese momento, para obtenerla, simplemente teclee pwd seguido de un return. Notará que ese es el lugar donde queda situado el usuario al ingresar al sistema, su directorio base o default(home directory).

Ejemplo:

```
$ pwd
/usr/acct/moises                    Ruta(absoluta) en la que se
                                    encuentra ubicado el usuario al
                                    entrar al sistema.
```

LISTADO DE SU DIRECTORIO

Para poder listar el contenido de su directorio se emplea el comando ls. Este comando tiene una serie de argumentos para desempeñar diferentes funciones. La mas común de ellas es el ejecutar el comando "ls -l" seguido de un return. El comando debe ser tecleado en minúsculas y con los espacios indicados. De contar con información en su directorio aparecerán una serie de datos, explicados posteriormente.

Ejemplo:

```
$ ls -l
  total 7
  -rwxr-xr-x  1 moises          64 Sep 12  1990 backup
  drwxr-xr-x  2 moises         112 Apr 30 10:35 berk
  drwxr-xr-x  2 moises         192 Apr 30 10:37 blast
```

```
drwxr-xr-x  2 moises          96 Apr 30 10:36 cables
-rw-r--r--  1 moises         601 Apr 30 15:42 calendar
drwxr-xr-x  2 moises          64 Apr 30 10:39 ciadeluz
drwxr-xr-x  3 moises         272 Apr 30 10:36 compresion
```

CAMBIO DE DIRECTORIO

Este se efectúa mediante el empleo del comando cd. Este comando recibe como argumento la ruta donde el usuario desea aparecer. Si el usuario teclea una ruta equivocada el sistema responde con un error, de lo contrario se dirige a ese directorio y no indica ningún resultado.

Ejemplo:

```
$ cd /tmp                    Comando para cambiar al directorio
$                            tmp bajo /. El sistema no envia
                             ningún mensaje.
```

LA RUTA ABSOLUTA

•

Esta ruta es la que posee cada archivo dentro del sistema desde el directorio raíz (o root) hasta el lugar donde se encuentra. En otras palabras, la ruta absoluta es la forma de indicar dentro del sistema la ubicación de un archivo, no importando el lugar dentro de la estructura de archivos donde se encuentre algún usuario.

LA RUTA RELATIVA

••

Esta ruta es la que tiene cualquier archivo desde el directorio de trabajo de cualquier usuario. Una ruta relativa, en términos generales es aquella que no inicia con el símbolo de diagonal.

EL DIRECTORIO DE CASA

Es el directorio que el usuario tiene desde el momento en el que ingresa al sistema. Normalmente es la cuenta que reside bajo usr, cuyo directorio es el nombre de ese usuario dentro del sistema. Cada vez que se teclea el comando cd sin argumentos, el usuario regresa esté donde esté a su directorio de casa.

Ejemplo:

```
$ cd /usr/spool              Cambia al usuario a /usr/spool
$ pwd                        Certifica donde se encuentra
/usr/spool
$ cd                         Regresa al directorio de casa del
$ pwd                        usuario.
/usr/acct/moises
```

COMANDO	*DEFINICIÓN*	*SINTAXIS*	*EJEMPLO*
cal	Despliega el calendario de un mes y año	cal <número_mes> <número_año>	cal 4 1999 /despliega el mes de abril de 1999
date	despliega la fecha actual	date	despliega "Mon Oct 18 13:42:38 1999
clear	Limpia pantalla	clear	
man	Despliega información sobre el comando tecleado	man <nombre_comando>	man date
cd	cambia de directorio	cd .. cd <nombre_directorio> cd /	Pasa al directorio superior Pasa dentro del directorio tecleado Pasa a la raíz del sistema
history	Lista los comandos previamente ejecutados	history <número_comandos>	history 10 /Lista los últimos diez comandos ejecutados
logout	Permite salirse de sesión	logout	
login	Permite identificarse ante el sistema	login <nombre_usuario>	login alumno
passwd	Cambia el password	passwd	
who	Lista los usuarios actualmente conectados al sistema	who	who am i /Indica con que nombre estoy identificado en el sistema
finger	Lista la información detallada acerca de los usuarios conectados actualmente al sistema.	finger	
cat	Concatena y despliega archivos	cat <nombre_archivo>	cat examen
cp	Copia el contenido de archivos y directorios	cp <archivo1> <archivo2> cp <archivo> <ruta>	cp prueba prueba1 cp prueba /usuarios
mv	Mueva y renombra archivos	mv <archivo1> <archivo2> mv <archivo> <ruta>	mv fin final mv fin /usuarios
diff	Compara y despliega diferencias entre archivos	diff <archivo1> <archivo2>	diff fin finales /Despliega línea por línea las diferencias entre fin y finales
more	Despliega el contenido de un archivo en pantalla	more <nombre_archivo>	more final
cmp	Compara el contenido de dos archivos	cmp <archivo1> <archivo2>	cmp fin final
file	Despliega las características del archivo	file <nombre_archivo>	file fin
touch	Crea un archivo vacío	touch <nombre_archivo>	touch vacio
head	Despliega las primeras n líneas de un archivo	head -<número_lineas> <archivo>	head –10 prueba
tail	Despliega las últimas n líneas de un archivo	tail -<número_lineas> <archivo>	tail –5 prueba

COMANDOS	*DEFINICIÓN*	*SINTAXIS*	*EJEMPLO*
sort	Ordena alfabéticamente líneas de un archivo y escribe la salida sobre la pantalla o un archivo	sort –o <archivo_salida> <archivo_original> sort –r <nombre_archivo>	sort –o ordena final sort –r final /ordena inversamente
rm	Borra un archivo	rm <nombre_archivo>	rm final
chmod	Permite modificar los modos de permisos de un archivo o directorio	chmod <clase> <operación> <permisos> <nombre_archivo>	chmod u = w permisos
ln	Crea ligas entre archivos	ln <archivo1> <archivo2>	ln fin final /Crea una liga entre fin y final
pwd	Indica el directorio actual	pwd	
mkdir	Crea un directorio	mkdir <nombre_directorio>	mkdir curso
ls	Lista los archivos del directorio	ls ls –l	/Lista los nombres de los archivos del directorio /Lista los archivos, fecha,, hora, permisos, propietario, nombre del archivo
rmdir	Borra el directorio	rmdir <nombre_directorio>	rmdir directorio
echo	Despliega un mensaje en la pantalla	echo <Mensaje>	echo 'Hola'
grep	Busca texto o palabras en un archivo	grep <palabra_a_buscar> <archivo>	grep hola final
wc	Lista el número de líneas, palabras, caracteres en un archivo	wc –l wc –w wc –c	Cuenta Líneas Cuenta palabras Cuenta caracteres
which	Localiza un comando y muestra su ruta	which <comando>	which vi /despliega la ruta:/usr/ucb/vi
vi	Ejecuta el editor de texto del sistema	vi <archivo>	vi final
du	Da la cantidad de kilobytes usado en un directorio y todos sus subdirectorios o de un archivo	du –a <nombre_directorio> du –x <nombre_archivo>	Despliega la cantidad de kilobytes usada por un directorio y su contenido Despliega la cantidad de kilobytes de un archivo

EL EDITOR VI

INTRODUCCIÓN

El vi (visual) es un editor de textos eficaz (aunque críptico), interactivo y orientado visualmente. El vi aprovecha toda la pantalla del terminal para desplegar el texto que se está editando. Al usar VI, no es necesario hacer referencia a las líneas por sus números, puede ubicarse el cursor en forma manual en cualquier línea o carácter. El VI lleva un registro de lo que está en pantalla y la limpia sólo cuando es indispensable. Este manejo de la pantalla permite al VI desplegar los cambios introducidos en el texto de la manera más eficiente posible y reducir el tiempo de respuesta, en especial con usuarios que acceden al sistema mediante líneas telefónicas lentas.

El VI no es un programa de formato de texto. No justifica márgenes, ni centra títulos, ni tiene las características de un sistema de procesamiento de textos.

MODOS DE OPERACIÓN

VI es parte de otro editor llamado EX e implica a dos de los cinco modos de operación de EX, el modo de mandato y el modo de inserción. En el modo de mandato, VI acepta los tecleados como mandatos, y responde a todos los mandatos a medida que se introducen. En el modo de inserción, VI acepta como texto de tecleados, desplegando el texto conforme se introduce.

Al comienzo de una sesión de edición, VI se encuentra en el modo de mandato. Hay varios mandatos, como insertar y agregar, que colocan a VI en el modo de inserción. Cuando se presiona la tecla ESC, VI siempre regresa al modo de mandato.

Los mandatos cambiar y reemplazar combinan los modos de mandato y de inserción. El mandato cambiar borra el texto que se desea cambiar y coloca a VI en el modo de inserción para poder introducir texto nuevo. El mandato reemplazar borra el carácter o se sobreescribe e inserta el o los que se ingresan.

EDICIÓN

En esta sección se describe cómo llamar a VI, introducir texto y salir de VI. Todos los mandatos de VI son de efecto final inmediato; no es necesario oprimir RETURN para indicar el final de un mandato.
Cuando se le da VI un mandato, es importante distinguir entre letras mayúsculas o minúsculas.

LLAMADA A VI

Para crear en el directorio de trabajo un archivo denominado práctica se llama a VI con la línea de mandato siguiente.

$VI práctica

El archivo práctica es nuevo; todavía no tiene texto. VI despliega uno de los mensajes siguientes en la línea de estado (en la parte inferior) del terminal para indicar que se está creando y editando un archivo nuevo.

"práctica" No such file or directory.

o bien

"práctica" ERROR

Cuando se edita un archivo existente, VI despliega las primeras líneas del archivo y da información del estado de éste en la línea de estado.

INTRODUCCIÓN DE TEXTO

Colocación de VI en el modo de inserción. Una vez obtenido el acceso a VI. colóquese en el modo de inserción oprimiendo la tecla i. VI no emite ninguna señal para indicar que se encuentra en el modo de inserción.

Si no se tiene la seguridad de estar en el modo de inserción, presiónese la tecla ESC; VI regresará al modo de mandato si se encontraba en el modo de inserción o emitirá un aviso (un sonido agudo o una luz) si se encontraba ya en el modo de mandato. Puede regresar a VI al modo de inserción oprimiendo i de nuevo.

Introducción de texto. Mientras VI está en el modo de inserción, puede ponerse texto en el buffer de trabajo escribiendo en el terminal. Si el texto no aparece en la pantalla conforme se escribe, es porque no se está en el modo de inserción.

Introdúzcase el párrafo modelo que se muestra en la pantalla sig., presionando la tecla RETURN para terminar cada línea. Al introducir texto, hay que cuidar algunos detalles: impedir que las líneas de texto vuelvan del lado derecho de la pantalla, al izquierdo, oprimiendo la tecla RETURN antes de que el cursor llegue al final del extremo derecho. Hay que asegurarse también de no acabar una línea con un espacio, pues algunos mandatos VI se comportan en forma extraña cuando encuentran una línea que termina con un espacio.

modelo
VI (visual) es un editor de textos eficiente (aunque críptico), interactivo, orientado visualmente. VI aprovecha la pantalla completa del terminal desplegando el texto que se está editando.
~
~
~
Cuando se detecta un error en la línea que se está introduciendo, puede corregirse antes de continuar. Véase el párrafo siguiente. Más adelante pueden corregirse otros errores. Al terminar de introducir el párrafo, se oprime la tecla ESC para devolver VI al modo de mandato. La pantalla se verá como el modelo mostrado anteriormente.

Corrección de texto conforme se inserta. Las teclas que permiten retroceder y corregir una línea de mandato del Shell (por lo común CTRL-H, @ y #) realizan la misma función cuando VI se encuentra en el modo de inserción. Además, puede utilizarse CTRL-W para retroceder sobre palabras. VI puede no eliminar texto de la pantalla al retroceder sobre éste. Sin embargo, el texto es suprimido del buffer de trabajo.

Hay dos restricciones al uso de estas teclas de corrección. Sólo se toleran retroceder sobre texto en la línea que se está introduciendo (no se puede retroceder a una línea anterior) y sólo se harán sobre texto recién introducido. Como ejemplo, supongamos que se está en el modo de inserción introduciendo texto y se oprime la tecla ESC para devolver VI al modo de mandato. Ahora ya no es posible retroceder sobre el texto introducido la primera vez que se utilizó en el modo de inserción aunque el texto se encuentre en la línea actual.

TERMINACIÓN DE LA SESIÓN DE EDICIÓN

Puede concluirse la sesión de edición en una u otra de las formas siguientes: conservando los cambios realizados durante la sesión o sin conservarlos. En general se desea conservarlos.

Terminación normal. La terminación normal de una sesión de edición requiere que VI grabe el texto editado (el contenido del buffer de trabajo) antes de regresar el control al Shell. Esta forma de concluir una sesión de edición asegura que el archivo de disco refleje cualquier cambio realizado.

> Hay que asegurarse que VI se encuentra en el modo de mandatos y utilizar el mandato ZZ (deben ser mayúsculas) para escribir el texto recién introducido, desde el buffer de trabajo hasta el disco y terminar la sesión de edición. La única ocasión en que no debe usarse el mandato ZZ para concluir una sesión de edición es cuando no desea almacenar el texto editado.

Después de dar el mandato ZZ, VI despliega el nombre del archivo que se está editando y el número de caracteres en el archivo; después devuelve el control al Shell.

Terminación anormal. Algunas veces es necesario terminar una sesión de edición sin grabar el contenido del buffer de trabajo. Cuando se utiliza el mando :q! RETURN (el símbolo : mueve el cursor a la línea de estado) para concluir una sesión de edición, no se conserva nada del trabajo de la sesión de edición actual; el contenido del buffer de trabajo se pierde. La próxima vez que se edite o utilice el archivo, este aparecerá como era antes de empezar la sesión de edición actual. Este mandato ha de utilizarse con precaución.

MOVIMIENTO DEL CURSOR

Mientras VI está en el modo de mandato, puede colocarse el cursor encima de cualquier caracter de la pantalla. También pueden desplegarse en ésta distintas partes del buffer de trabajo. Manipulando la pantalla y la posición del cursor, éste puede situarse sobre cualquier caracter del buffer de trabajo.

movimiento del cursor por unidades de medida

MANDATO	MUEVE EL CURSOR
Espacio, flecha derecha	un espacio a la derecha
h o flecha izquierda	un espacio a la izquierda
w	una palabra a la derecha
W	una palabra delimitada por blancos a la derecha
B	una palabra a la izquierda
B	una palabra delimitada por blancos a la izquierda
$	fin de línea
O	principio de línea
RETURN	principio de siguiente línea
j o flecha descendente	hacia abajo una línea
K o flecha ascendente	hacia arriba una línea
)	fin de frase
(	principio de frase
}	fin de párrafo
{	principio de párrafo
}}	fin de archivo

MODO DE INSERCIÓN

Los mandatos de inserción, adición de texto, abrir líneas y reemplazar, colocan a VI en el modo de inserción. Mientras se encuentra en eso modo VI, puede ponerse texto nuevo en el buffer de trabajo. Al terminar de introducir texto, para devolver VI al modo de mandato, siempre se pulsa la tecla ESC.

El mandato de inserción

El mandato i coloca a VI en el modo de inserción y coloca el texto introducido antes del carácter sobre el cual se encuentra en el cursor (el caracter actual). Aunque el mandato i algunas veces escribe sobre el texto de la pantalla, éste reaparece el presionar ESC y devolver a VI al modo de mandato. Se utiliza el mandato i para insertar unos cuantos caracteres o palabras en un texto ya existente o para insertar texto en un nuevo archivo.

Los mandatos de adición (append)

El mandato a es similar al i, exepto en que pone el texto introducido después del carácter actual. El mandato A coloca el texto después del último caracter de la línea en curso.

Los mandatos de apertura (open)

Los mandatos o y O abren una línea en blanco dentro del texto existente, colocan el cursor al principio de la línea nueva (en blanco) y sitúan a VI en el modo de inserción. El mandato O abre una línea sobre la línea en curso; o la abre abajo. Se utilizan mandatos Open para introducir líneas nuevas en un texto ya existente.

Los mandatos de reemplazar (replace)

Los mandatos R y r hacen que el nuevo texto introducido sobreescriba (o reemplace) al existente. El caracter que sigue a un mandato r escribe sobre el caracter en curso. Después de ese caracter, VI regresa de forma automática al modo de mandato, sin necesidad de oprimir la tecla ESC.
El mandato R hace que todos los caracteres subsecuentes reemplacen el texto existente hasta pulsar ESC y devolver a VI al modo de mandato.

MODO DE MANDATO: BORRADO Y CAMBIO DE TEXTO

El mandato deshacer (undo)

El mandato deshacer, o u, deshace lo que acaba de hacerse. Restaura texto borrado o cambiado por error. El mandato Undo sólo arregla el último texto borrado. Si se borra una línea y después se cambia una palabra, el mandato sólo restaura la palabra cambiada, no la línea borrada. El mandato U restaura la línea actual a la forma en la que estaba antes de empezar a cambiarla, aunque se hayan realizado muchos cambios.

El mandato borrar un carácter (delete character)

El mandato x borra el caracter en curso. Si este mandato va seguido de un factor de repetición, entonces pueden borrarse varios caracteres de la línea actual, comenzando con el caracter actual.

El operador borrar (delete)

El operador d elimina texto del buffer de trabajo. La cantidad de texto que d suprime depende del factor de repetición y de la unidad de medida que se indican después de introducir d. Después de borrar el texto, VI se encuentra en el modo de mandato.
Advertencia: El mandato d RETURN, en forma ilógica, borra dos líneas, la línea en curso y la siguiente. Para borrar sólo la línea en curso se utiliza el mandato dd, o se antepone a dd un factor de repetición para borrar varias líneas.

BÚSQUEDA DE UNA CADENA

Los mandatos de búsqueda (search)

VI buscará por el buffer de trabajo una cadena de texto específica. Para encontrar la siguiente ocurrencia de una cadena (hacia adelante), oprímase la tecla diagonal (/),

palabra	Una secuencia de letras y/o números
PALABRA	Una secuencia de caracteres incluyendo los espacios que siguen
arch	Algún archivo del disco (Existente o no)
patrón	Secuencia de caracteres a utilizar en un patrón de búsqueda
Movimiento	Algún comando de movimiento

EDITORES DE TEXTO
ED

%ed nombre_archivo

COMANDOS

i insertar
a add
w escribir
q quit

Modo edición
Modo comando

a enter
Hola (modo adición)
. enter
(modo comando)
w enter (modo comando)
n numero de bytes
q enter (me saca al prompt)

VI

i insertar escribe antes del carácter marcado
a add escribe después del carácter marcado
A add fin de linea
o siguiente linea abajo
O siguiente linea arriba
r reemplaza
R reemplaza hasta ESC (reemplaza de allí hasta que se de ESC)
h izquierda k arriba j abajo l derecha
x borra carácter
d borra linea 5d 5 lineas
G ir a linea
$ fin de linea
0 inicio de linea
ESC. enter:
w write w [otro nombre.txt] write&exit ~ wq
x write y exit X encriptar
q quit
ESC cambiar de modo

CONFIGURACIÓN DE ARCHIVOS DE RED

nsswitch.conf — archivo que contiene asignado el DNS, en la linea que contiene la palabra hosts, debe estar
hosts: dns

netconfig — archivo de configuración del protocolo tcp de la red
tcp tpi_cots_ord v inet tcp /dev/tcp -

netmasks — archivo que asocia la dirección IP con la de la marcara de red
132.248 170.22 255.255.255.0

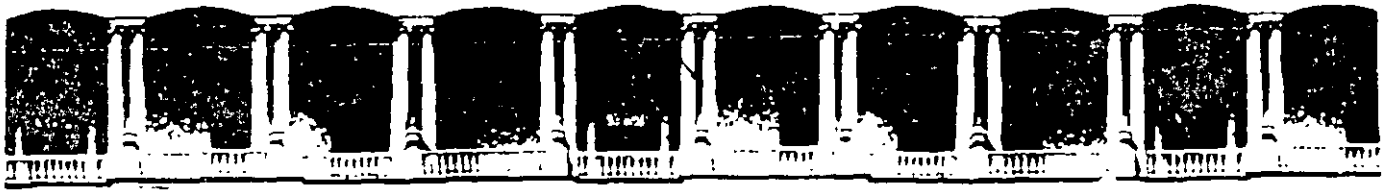
networks — archivo asociado con números y nombres de red
sintaxis:
network-name network-number nicnames
loopback 127
arpanet 10 arpa #historial

defaultrouter — archivo que define la dirección IP del ruteador
132.248 170.254

hostname.hme0 — archivo que define el nombre del host y el dominio al que pertenece
opera2.dgsca unam mx

resolv.conf — archivo que define las direcciones de servidores DNS externos
domain dgsca.unam.mx
nameserver 132 248.204.1
nameserver 132.248 10.2

hosts — archivo que define la dirección IP del host
127 0 0.1 localhost
132.248 170 22 opera2 dgsca.unam.mx
loghost



FACULTAD DE INGENIERIA U.N.A.M.
DIVISION DE EDUCACION CONTINUA

INTRODUCCION A UNIX

C O M P L E M E N T O

MARZO DEL 2002

Palacio de Minería Calle de Tacuba 5 Primer piso Deleg. Cuauhtémoc 06000 México, D F APDO Postal M-2285
Teléfonos: 5512-8955 5512-5121 5521-7335 5521-1987 Fax 5510-0573 5521-4021 AL 25

ENTRADA Y SALIDA ESTANDAR

Como hemos visto una de las características del sistema UNIX. es la forma general y flexible de tratar los archivos y la facilidad con la que se puede controlar el lugar desde donde los programas obtienen la entrada y envían la salida. La mayor parte de las ordenes aceptan entrada desde teclado. desde un archivo almacenado o desde la salida de una orden.

Este planteamiento flexible para la entrada y la salida. se basan en los conceptos fundamentales del sistema Unís de entrada y salida estándar, o I/O estándar.

En la siguiente figura se muestra una orden que acepta entrada y produce salida utilizando I/O estándar. La orden obtiene su entrada a través del canal etiquetado "entrada estándar y produce su salida a través del canal etiquetado "salida estándar". La entrada puede provenir de un archivo del teclado o de una orden. La salida puede ir a un archivo. a la pantalla o a otra orden.

La orden no necesita saber de donde proviene la entrada o a donde va la salida. Es el shell el que establece estas conexiones con base en las instrucciones en la línea de comandos.

Una de las funciones más importantes del shell es la gestión de la entrada y salida estándar, de manera que usted solo necesita especificar de donde obtiene la entrada su orden y donde debe enviar la salida. Esto se realiza mediante el mecanismo de redirección I/O. que incluye la redirección de archivo y los causes.

Un ejemplo de redirección de archivo es la orden siguiente:

```
$ ls - l > Temp.
```

Un ejemplo de la características de cauce es la orden siguiente:

$ ls - 1 | lp

Esta utiliza un cauce ls a lp, a fin de imprimir una copia del dirección actual.

Lista de los símbolos utilizados para decir al shell donde obtener la entrada y donde enviar la salida.

A estos se le denomina operadores de redirección del shell.

Símbolo	Ejemplo	Función
<	cmd < file	Toma la entrada para cmd de file
>	cmd > file	Toma la salida de cmd a file
>>	cmd >> file	Añade la salida de cmd a file
|	cmd1 | cmd2	Ejecuta cmd1 y envia la salida a cmd2

REDIRECCIÓN DE SALIDA

Cuando introduce una orden, puede utilizar los operadores de redirección <, >, (léase flecha izquierda y flecha derecha) y >>, para decir al shell que redirija la entrada y la salida. Por ejemplo.

$ ls > Temp.

Hace que el shell mande la salida de ls al archivo Temp.. Si ya existe un archivo con ese nombre en el directorio actual, resulta sobrescrito -- su contenido se vacía y se remplaza por la salida de la orden --. Si no existe un archivo con el nombre que usted especifica el shell crea uno antes de ejecutar la orden.

El operador >> añade datos a un archivo sin eliminar los ya existentes. Para ilustrar la diferencia entre redirección y adición, considere los siguientes ejemplos:

```
$ cat conf > meetings
```

Esto copia el contenido de conf a meetings, sustituyendo el contenido anterior de meetings, si existe, y creando meetings si no existe. Compare éste con el siguiente:

```
$ cat conf >> meetings
```

Esto añade el contenido de conf al final del archivo denominado meetings, sin destruir ninguna otra información del archivo.

En cualquier caso, si meetings no existe, se crea y el contenido de conf se copia en el.

REDIRECCIÓN DE LA ENTRADA

De la misma forma que se utiliza el símbolo de la flecha derecha, > , para redirigir la salida estándar, se utiliza el símbolo de flecha izquierda, <, para redirigir la entrada estándar. El símbolo < dice al shell que utilice al archivo siguiente como entrada estándar de una orden. Por ejemplo la orden siguiente imprime el contenido de file:

```
$ cat < file
```

El < dice al shell que ejecute cat con file como entrada estándar.

Muchas ordenes también proporcionan una forma para poder especificar un archivo de entrada directamente como argumento. Por ejemplo, cat permite nombra uno o más archivos de entrada como argumentos. Así, las ordenes.

```
$ cat < chap1
```

y

```
$ cat chap1
```

Estas opciones visualizan el mismo archivo sobre la pantalla.

Aunque el resultado es el mismo, el mecanismo subyacente es distinto. En el primer caso, el shell conecta a chap1 a la entrada estándar que lee cat. En el segundo caso, la orden cat abre chat1 y lee de el su entrada.

UTILIZACIÓN DE LA ENTRADA ESTANDAR CON ARGUMENTOS DE NOMBRES DE ARCHIVOS

Varias ordenes proporcionan la forma para combinar la entrada estándar con los argumentos de nombres de archivos. Para ello. se utiliza un signo menos. - , como argumento que indica que la orden deberá tomar su entrada de la entrada estándar. Resulta a veces útil hacer referencia explícitamente a la entrada o salida estándar en una línea de orden. Por ejemplo, cuando se quiere mezclar entrada procedente del teclado con entrada procedente de un archivo.

Por ejemplo se puede particularizar una carta de formulario combinando información tecleada tal como el nombre del destinatario. con el texto almacenado:

```
$ cat - form_letter > salida


Querida sue,
CRTL-D
```

El ejemplo anterior concatena la entrada del teclado (entrada estándar) con el contenido de form_letter. Lee todo lo que se teclea hasta CTRL-D. que indica el final de la entrada por teclado. Este uso de " - " para representar la entrada estándar. no es seguido por todas la ordenes. SVR4 dispone de otra forma para especificar directamente la entrada estándar. a través del nombre del archivo lógico /dev/stdin./dev/stdin siempre hace referencia a la entrada estándar. Usted puede utilizarlo siempre que quiera incluir explícitamente la entrada estándar en una línea de orden. Usándolo. el ejemplo anterior se convierte en

```
$ cat  /dev/stdin form_letter > salida
```

REDIRECCIÓN DE LA ENTRADA Y LA SALIDA.

Los ejemplos anteriores han mostrado como se puede redireccionar la entrada o la salida. La entrada y la salida se pueden redireccionar simultáneamente. como ocurre en el ejemplo que utiliza la orden sort. para clasificar la información de file1 y colocarla en file2:

```
$ sort < file1 > file2
```

Esta orden utiliza los operadores de redirección. para tomar la entrada de sort de file1 y colocar la salida de sort en file2. El orden en que se indica la entrada y la salida de archivos. es indiferente. En el ejemplo siguiente :

```
$ sort > file2 > file1.
```

El efecto es el mismo.

UTILIZACIÓN DE TUBERIAS.

El cauce es otra forma de redirección de la salida proporcionada por el shell. El símbolo del cauce, | , dice al shell que tome la salida estándar de una orden y la utilice como entrada estándar de otra orden.

Esta posibilidad de utilizar cauces para unir ordenes individuales y ejecutar así secuencias de operaciones, es una de las características más notables del sistema UNÍS. Se dispone así, de una forma fácil de encadenar ordenes simples a fin de componer una función más compleja.

EL SHELL

Una gran parte del uso del sistema UNIX consiste en emitir órdenes. Cuando se emite una orden en la línea de comandos de UNIX el usuario se esta relacionando con el shell. la parte del sistema UNIX a través de la cual se controlan los recursos de sistema operativo UNIX. El shell proporciona muchas de las características que hacen al sistema UNIX un entorno potente y flexible. Se trata de un interprete de órdenes. un lenguaje de programación y más. Como interprete de ordenes, el shell lee las órdenes que se le introducen y dispone de lo necesario para que se ejecuten.

Además se puede utilizar el lenguaje de órdenes del shell como un lenguaje de programación de alto nivel para crear programas denominados guiones.

Nosotros podemos obtener otras versiones de shell, tales como ***bash***, ***csh***, ***tcsh*** y ***zsh*** de alguna ubicación de Internet. Los conceptos y características descritos en este capítulo también se aplican al shell Korn, y la mayor parte de ellos se aplica al shell C.

El sistema operativo UNIX también proporciona otros shell, el shell C (***csh***). el shell Korn (***ksh***). que ofrece varias ampliaciones valiosas al shell de sistema UNIX. Existen también otros shell que se utilizan ampliamente que no están incluidos en el las versiones estándar de UNIX.: hasta el momento se encuentra UNIX en la versión V. identificándose SVR4 (System Five. Release 4).

Estas están incluidas en versiones especiales de UNIX: estos incluyen:

- ✓ ***bash***. parte del proyecto GNU. que sigue la sintaxis del shell estándar de SVR4, incorporando muchas características del shell C.
- ✓ ***tsh***. sigue la sintaxis del shell C y proporciona funcionalidades añadidas.
- ✓ ***zsh***. se asemeja al shell Korn. aunque no es compatible completamente con el; contiene muchas ampliaciones incluyendo características del shell C.

SHELL DE INCIO DE SESIÓN.

Cuando se establece una sesión en un sistema UNIX se inicia automáticamente un programa de shell. Este es el shell de inicio de sesión. El programa de shell particular que se ejecuta cuando usted se presenta. esta determinado por su entrada en el archivo /etc/passwd. Este archivo contiene información que el sistema necesita conocer sobre cada usuario. incluyendo

el nombre, el ID de inicio de sesión, etc. El último campo de este archivo contiene el nombre del programa a ejecutar, que es su shell de inicio de sesión.

INCIACIÓN DEL SHELL Y SU PROFILE

Cuando se inicia su shell de inicio de sesión, busca un archivo llamado .profile, el cual contiene ordenes que particularizan el entorno de inicio de sesión. Así, el shell lee el archivo y lleva a cabo la instrucciones que contiene.

El archivo .profile es un ejemplo sencillo de guión de shell. El contenido del archivo .profile son a su vez ordenes e instrucciones para el shell. Este archivo contiene típicamente, información que el shell y otros programas necesitan conocer para particularizar el entorno de trabajo, como si se va a trabajar en entorno gráfico, el tipo de shell que se utilizará por default para ese usuario en cuestión, directorio **HOME** del usuario, privilegios de usuario especiales, etc.

QUE HACE EL SHELL

En general, una línea de orden contiene un nombre y unos argumentos. A excepción de ciertas palabras clave como for y while, cada línea de orden finaliza con un NEWLINE, que es el término del sistema UNIX para el carácter que se produce cuando se teclea ENTER. El shell no comienza a procesar la línea de orden hasta que recibe un ENTER.

Los argumentos de la línea de orden son opciones que modifican lo que hace una orden y cómo lo hace, y la información que necesita, como puede ser el nombre de un archivo de que obtener datos. Las opciones normalmente se indican, aunque no siempre con un signo -.

A menudo la línea de orden incluye argumentos y símbolos que son realmente instrucciones para el shell. Por ejemplo, cuando se utiliza el símbolo > para dirigir la salida de una orden a un archivo, o el símbolo de cauce, | , para utilizar la salida de una orden como entrada a otra, o el ampersand, &, para ejecutar una orden en modo subordinado, realmente se están dando instrucciones al shell. Este capítulo explicara como procesa el shell estas y otras instrucciones de la línea de orden.

VARIABLES DEL SHELL

El shell dispone de un mecanismo para definir variables que se pueden utilizar para contener información usada por los programas del sistema o para uso propio. Algunas variables las utiliza el propio shell y otros programas del sistema UNIX. Se pueden definir otras para uso

propio. Las variables de shell se pueden utilizar para personalizar o particularizar la información relativa a nombres de directorios y de archivos que necesitan programas y para particularizar la forma en la que los programas (incluyendo el propio shell) interactuan con el usuario.

VARIABLES DEL SHELL COMUNES.

Lo siguiente es un resumen breve de algunas de las variables de shell más comunes, incluyendo aquellas que filja automáticamente el sistema. Son las denominadas variables de entorno ya que configuran distintos aspectos del entorno de trabajo del entorno del usuario.

HOME contiene el nombre de camino absoluto de su directorio de trabajo del usuario en cuestión. HOME es definida automáticamente y fijada a su directorio de trabajo como parte del proceso de inicio de sesión. El propio shell utiliza esta información para determinar el directorio al que cambiar cuando usted teclea la orden cd sin argumento. PATH, lista, en orden los directorios en los que el shell busca para encontrar el programa a ejecutar cuando usted teclea una orden. PATH contiene una lista de nombres de directorios, separados por dos puntos. Un **PATH**, por omisión es definido por el sistema, pero la mayor parte de los usuarios lo modifican para añadir directorios de ordenes adicionales. Un campo vacío en la cadena **PATH** significa buscar en el directorio actual. Un ejemplo típico de **PATH** particularizado, en este caso para el usuario YOU, es el siguiente:

```
PATH=/bin:/home/you/bin:/var/add-on/bin:
```

Este valor de **PATH** significa que cuando usted introduzca una orden, el shell buscará en primer lugar el programa en el directorio /bin: despues en el subdirectorio bin del directorio de trabajo del usuario: a continuación buscara en /var/add-on/bin, y finalmente en el directorio actual (indicado por el campo vacío al final).

CDPATH es similar a **PATH**. Lista en orden los directorios en los que busca el shell para encontrar un subdirectorio a cambiar cuando usted utiliza la orden cd. Los directorios que busca el shell se listan de la misma forma que los directorios de su **PATH**. Si el valor de **CDPATH** es

```
CDPATH=:home/you:/home/you/proyects:/home/sue
```

entonces cuando usted ejecuta la orden,

```
$ cd Book
```

cd busca en primer lugar el directorio denominado Book en el directorio actual (indicado por la entrada vacía antes del primer :); después en /home/you; luego en su directorio proyects, y finalmente en el directorio de trabajo sue.

PS1 y **PS2** definen los signos de sus peticiones de orden primaria y secundaria, respectivamente. Sus valores por omisión son $ para **PS1** y > para **PS2**.

LOGNAME contiene su nombre de usuario, lo fija automáticamente el sistema.

MAIL contiene el nombre del directorio en el que se coloca su correo nuevo. El shell utiliza esta variable para notificarle cuando se añade una nueva información a su directorio.

SHELL contiene el nombre del programa de shell utilizado por el usuario.

VARIABLES DE SHELL COMUNES

Variable del shell	Descripción	Ejemplo	Notas
CDPATH	Lista de directorios buscados por la orden cd	:/home/curso_unix:/home/curso_unix/libro	Fijado por el usuario
HOME	Nombre de camino de su directorio de trabajo	:/home/curso_unix	Fijada automáticamente en el inicio de sesión
LOGNAME	Su nombre de usuario	curso_unix	Fijada automáticamente en el inicio de sesión
MAIL	Nombre del camino del directorio que contiene su correo	curso_unix	/home/curso_unix/Mail
MAILFILE	Archivo que contiene el correo nuevo para mailx	:/var/mail/$LOGNAME	Utilizada por mailx en el archivo .mailrc
PATH	Lista de directorios donde el shell busca las órdenes	:/bin:home/curso_unix/bin	Fijada automáticamente en el inicio de sesión
PS1	Indicativo primario del shell	Sistema 1$	por omisión es $
PS2	Indicativo primario del shell	=+>	Por omisión es >

SHELL	Nombre del camino de su shell	/bin/ksh	Fijada automáticamente
TERM	Define su tipo de terminal para vi y otras órdenes orientadas a pantalla	vt100	Fijada por el usuario; sin valor por omisión.
TZ	Información de la zona horaria		

DIFERENCIAS ENTRE SHELL

Dada la estandarización del shell C y Shell Korn, trataremos únicamente estas dos versiones más ampliamente utilizadas.

El shell C y el Shell Korn se desarrollaron para disponer de características y capacidades adicionales que no ofrecía el **sh**. Tanto **csh** como **ksh** proporcionan un número importante de mejoras respecto al shell. Entre ellos están la edición de línea de orden, que posibilita la edición de las líneas de órdenes cuando se introducen; la lista de la historia de órdenes, que le permiten revisar las órdenes que se han utilizado en una sesión y los alias de órdenes, que se pueden utilizar dar nombres más adecuados para las órdenes. **csh** y **ksh** también proporcionan la posibilidad de utilizar órdenes desde una lista histórica para simplificar la creación de nuevas órdenes, la protección de sobre escritura accidental en archivos existentes cuando se redirige la salida hacia ellos, un conjunto de características útiles y capacidades extendidas de programación del shell.

El shell C fue desarrollado por Bill Joy como parte del sistema **UNIX** de Berkeley. Fue el primero de los shell mejorados. Proporciona las características del shell de Bourne, además de un conjunto de características nuevas y ampliaciones de las antiguas. Las características mas importantes del shell incluyen el control de los trabajos, las listas históricas, alias y una sintaxis estilo lenguaje C, como una extensión de sus capacidades para programar.

El shell Korn, **ksh**, es otra alternativa popular a **sh**. Fue desarrollado en 1982 por David Korn de los Laboratorios Bell, y ha evolucionado a través de una serie de nuevas versiones que iban aumentando en potencia. El shell Korn proporciona un súper conjunto de características altamente compatibles con el shell de Bourne. Añade sus propias versiones de la mayoría de mejoras que se encuentran en el shell C, así como muchas otras características potentes, conservando la sintaxis y características básicas de sh. Proporciona un mecanismo potente de historia y un formato de edición de línea de órdenes que es altamente compatible con los populares editores de texto **vi** y **ed**. Una ventaja importante de ksh con respecto a **csh** es que los programas de shell escritos para **sh** se ejecutan generalmente sin modificaciones bajo **ksh**, mientras que requieren de ciertos cambios para poder ser ejecutados bajo **csh**.

OBTENCIÓN DEL VALOR DE UNA VARIABLE DE SHELL

Además de fijar valores, a veces usted necesita obtener el valor de una variable de shell. Por ejemplo, puede querer ver si su actual valor PATH incluye un directorio particular. O bien, puede querer utilizar el valor de una variable de shell se escribe con letras mayúsculas, por ejemplo: PATH. El valor de la variable se obtiene mediante el nombre precedido por el signo dólar, $. El valor de PATH ES $PATH.

Para obtener el valor de una variable de shell preceda el nombre de la variable con un signo dólar, $. Cuando el shell lee una línea de orden, interpreta cualquier palabra que comienza con $ como una variable y la sustituye por su valor.

Para ver el valor de una variable puede utilizar la orden echo. Esta orden repite (imprime) su entrada estándar en su salida estándar. Por ejemplo:

```
$ echo hola
hola
```

Cuando emplee echo para obtener el valor de una variable, escriba el nombre de la variable precedido por $ como argumento de echo. Por ejemplo, $ echo $PATH la siguiente línea de orden visualiza su PATH actual:

```
$ echo $PATH
/bin:/home/you/bin:/var/add-on/bin:
```

Se puede utilizar el valor de la variable HOME en las órdenes para evitar teclear el nombre de camino completo. Lo siguiente mueve el archivo notes al directorio Stuff en su directorio de trabajo:

```
$ mv notes $HOME/Stuff
```

Se utiliza $HOME como forma más conveniente de especificar parte del nombre de camino del directorio destino.

Su puede usar la orden set par ver todas las variables actuales del shell y sus valores. Por ejemplo:

```
$ set
CDPATH=:/home/sue:/home/sue/db:/home/sue/progs
HOME=/home/sue
MAIL/=/var/mail/sue
MAILCHECK=30
MAILPATH=/var/mail/sue:/home/sue/rje
MAILRC=/home/sue/mail/.mailrc
MBOX=/home/sue/mail/mbox
PATH=usr/bin:/usr/lbin:home/sue/bin
PSl=:
```

```
PS2=...
SHELL=/usr/bin/sh
TERM=AT386-M
TZ=EST5EDT
```

La mayor parte de las variables de la lista anterior se trataron anteriormente. En este ejemplo, PS1 Y PS2 han sido redefinidos. MAILRC es un ejemplo de variable utilizada por una orden particular –en este caso la orden mailx-.

DEFINICION DE LAS VARIABLES DE SHELL

Aunque HOME, PS1, PS2 y otras variables comunes son establecidas automáticamente por el sistema, a otras no les ocurre esto. Usted debe definirlas utilizando la capacidad de definición de variables de shell. TERM y MAILFILE son ejemplos de variables de shell que no se definen automáticamente.

Una variable de shell se define escribiendo su nombre seguido por un signo = junto con su valor. Para fijar el valor de la variable de terminal a vt100, utilice la orden:

```
$ TERM=vt100
```

De la misma forma se pueden redefinir algunas de las variables actuales, como HOME y ps1. Por ejemplo, para cambiar el símbolo del indicativo de órdenes primario a +, puede teclear

```
$PS1=+
```

Siempre que se define una nueva variable o se particulariza una existente, no deben existir espacios entre el nombre de la variable, el signo = y el valor. El valor puede contener un espacio o incluso uno o más NEWLINE, pero si los contiene deben ir entre comillas. Por ejemplo, se puede definir un indicativo de ordenes que incluya dos palabras:

```
$PS1="hola:"
```

Si hace esto, el shell le hará peticiones de la forma:

```
Hola:
```

Las variables comunes como HOME, PS1 y TERM se definen normalmente en el archivo .profile, pero como indican estos ejemplos, también pueden escribirse directamente desde el teclado. Si se redefine una variable escribiendo su nuevo valor, en lugar de colocarlo en una línea de .profile, el nuevo valor se mantiene durante la sesión actual, pero volverá a su antiguo valor la siguiente vez que usted realice la conexión. Para cambiarlo de forma permanente (o hasta que usted quiera cambiarlo de nuevo), tienen que colocar la nueva redefinición el profile.