



## CENTRO DE ESTUDIOS FINANCIEROS

VIRIATO, 52	28010 MADRID	914 44 49 20
PONZANO, 15	28010 MADRID	914 44 49 20
G. DE GRÀCIA, 171	08012 BARCELONA	934 15 09 88
ALBORAYA, 23	46010 VALENCIA	963 61 41 99

[www.cef.es](http://www.cef.es)

[info@cef.es](mailto:info@cef.es)

## Índice Tema 8

---

1. Historia.
  - 1.1. UNIX.
    - 1.1.1. Estandarización de UNIX.
  - 1.2. LINUX.
  - 1.3. WINDOWS.
2. UNIX, LINUX.
  - 2.1. UNIX.
  - 2.2. LINUX.
    - 2.2.1. Características del núcleo.
  - 2.3. Procesos y threads.
  - 2.4. Gestión de memoria.
  - 2.5. Entrada/salida.
  - 2.6. El sistema de archivos.
  - 2.7. Seguridad.
3. Familia WINDOWS.
  - 3.1. Visión general.
  - 3.2. Procesos y threads.
  - 3.3. Gestión de memoria.
  - 3.4. Entrada/salida.
  - 3.5. El sistema de archivos.
  - 3.6. Seguridad.

4. Los intérpretes de comandos.
  - 4.1. El SHELL.
  - 4.2. Historia de los SHELL de UNIX-LINUX.
  - 4.3. Comparativo entre SHELLs.
  - 4.4. El intérprete de comandos de WINDOWS.
5. Estructura de archivos.
  - 5.1. Los archivos de UNIX, LINUX.
  - 5.2. La jerarquía de directorios UNIX, LINUX.
  - 5.3. Los archivos de WINDOWS.
6. Interfaces gráficos.
  - 6.1. Los orígenes del GUI de UNIX, LINUX.
  - 6.2. Otros GUI de UNIX, LINUX.
  - 6.3. La interfaz hombre-máquina de WINDOWS.
7. Estado actual de los S.O.
  - 7.1. UNIX, LINUX.
  - 7.2. WINDOWS.



## CENTRO DE ESTUDIOS FINANCIEROS

VIRIATO, 52	28010 MADRID	914 44 49 20
PONZANO, 15	28010 MADRID	914 44 49 20
G. DE GRÀCIA, 171	08012 BARCELONA	934 15 09 88
ALBORAYA, 23	46010 VALENCIA	963 61 41 99

www.cef.es

info@cef.es

### TEMA 8

#### Características técnicas y funcionales de los sistemas operativos: Windows, Linux y Unix.

##### 1. HISTORIA.

###### 1.1. UNIX.

Unix es uno de los sistemas operativos más ampliamente usado en ordenadores, debido a su historia, que evoluciona en los Laboratorios Bell de AT&T con un simulador de un viaje espacial en el sistema solar, pasando por su expansión en universidades y la creación de las versiones más importantes que son la de la Universidad de Berkeley y el Sistema V de la misma AT&T.

UNIX funciona tanto en ordenadores personales como en mainframes, trabajando en sistemas uniprocador y multiprocador. Fue creado por un equipo del Laboratorio Bell de la AT&T a principios de los 70: Ken Thompson y Dennis Ritchie basado en el proyecto MULTICS, que había quedado abandonado.

Después de una primera versión, en ensamblador PDP-7 denominada UNICS que posteriormente pasó a llamarse UNIX, se rescribe en un lenguaje de alto nivel, llamado B. Posteriormente, Thompson y Ritchie desarrollan el C y lo vuelven a describir. En 1976 se difunden gratuitamente los fuentes de UNIX por las universidades de EE.UU. (Versión 6).

A continuación se muestra un resumen de la evolución del sistema UNIX, con sus hitos más significativos:

1978: Versión 7.

1978-8x: Versión de Berkeley (BSD) de UCB (University of California Berkeley), incorpora memoria virtual, utilidades, soporte de redes (TCP, sockets).

1983: System V de AT&T.

198x: Formación de consorcios para fijación de estándares: XPG, OSF, etc. Normas POSIX. IBM y DEC empiezan a emplear UNIX.

Finales de 198x: las versiones 4.3BSD y SystemV R3 son incompatibles.

199x: Linux (UNIX gratuito).

### 1.1.1. Estandarización de UNIX.

Debido a las múltiples versiones en el mercado de UNIX, se comenzaron a publicar estándares para que todas las versiones fuesen «compatibles». La primera de ellas la lanzó AT&T, llamada SVID (System V Interface Definition), que definiría cómo deberían ser las llamadas al sistema, el formato de los archivos y muchas cosas más, pero la otra versión importante, la de Berkeley (Berkeley Software Distribution o BSD) simplemente la ignoró. El primer intento serio de reconciliación se inició bajo los auspicios de la IEEE con un proyecto colectivo denominado POSIX de «Portable Operating System». Con la colaboración de personas de la industria, las universidades y el gobierno el comité POSIX produjo el estándar conocido como 1003.1 que define un conjunto de procedimientos de biblioteca que debe proporcionar todo sistema UNIX que cumpla con la norma. Posteriormente, los institutos ANSI e ISO se interesaron en estandarizar el lenguaje «C» y conjuntamente se publicaron definiciones estándares para otras áreas del sistema operativo como la interconectividad, el intérprete de comandos y otras. En la tabla 1 se muestran las especificaciones POSIX.

TABLA 1

### LAS ESPECIFICACIONES DE POSIX

ESTÁNDAR	DESCRIPCIÓN
1003.0	Introducción y repaso.
1003.1	Procedimientos de biblioteca
1003.2	Intérprete y comandos.
1003.3	Métodos de prueba.
1003.4	Extensiones para tiempo real.
1003.5	Lenguaje Ada.
1003.6	Extensiones para la seguridad.
1003.7	Administración del Sistema.
1003.8	Acceso transparente a archivos.
1003.9	Lenguaje Fortran.
1003.10	Supercómputo.

Al momento del auge de los estándares de POSIX, desgraciadamente se formó un grupo de fabricantes de computadoras (IBM, DEC y Hewlett-Packard) que lanzaron su propia versión de UNIX llamada OSF/1 (de Open Software Foundation). Lo bueno fue que su versión tenía como objetivo cumplir con todos los estándares del IEEE, además de un sistema de ventanas (el X11), una interfaz amigable para los usuarios (MOTIF) y las definiciones para cómputo distribuido (DCE) y administración distribuida (DME). La idea de ofrecer una interfaz amigable en UNIX no fue original de OSF, ya en la versión 3.5 de SunOS de Sun Microsystems se ofrecía una interfaz amigable y un conjunto de librerías para crear aplicaciones con interfaz gráfica técnicamente eficiente y poderosa llamada SunWindows o SunVIEW. Esta interfaz junto con sus librerías estaban evolucionando desde la versión para máquinas aisladas hacia una versión en red, donde las aplicaciones podían estar ejecutando en un nodo de la red y los resultados gráficos verlos en otro nodo de la red, pero Sun tardó tanto en liberarlo que le dio tiempo al MIT de lanzar el X11 y ganarle en popularidad.

AT&T formó, junto con Sun Microsystems y otras compañías, UNIX International y su versión de UNIX, provocando así que ahora se manejen esas dos corrientes principales en UNIX.

## 1.2. LINUX.

El sistema operativo Linux se genera inspirándose en dos sistemas operativos: el sistema abierto UNIX creado en 1969 por Ken Thompson y Dennis Ritchie en los laboratorios de Bell (de este sistema se toman sus características, especificaciones y funcionamiento), más el sistema educativo MINIX creado en 1987 por Andrew S. Tanenbaum (del cual se toma la estructura y código del núcleo). Con todo esto, en 1991 Linus Torvalds crea Linus's Unix = Linux Kernel; esto es, crea sólo el núcleo del sistema sin la capa de servidores, manejadores, aplicaciones gráficas, etc. que serán creadas posteriormente por otros autores. El código del núcleo lo podemos encontrar en la dirección ([www.kernel.org](http://www.kernel.org)). El núcleo actual tiene aproximadamente 1,5 millones de líneas de código, y representa menos del 50 por ciento de todo el código del sistema.

En la comunidad de programadores se crea el proyecto GNU (Gnu's Not Unix), proyecto para generar software libre, donde se generan editores Emacs, compiladores gcc, interprete de comandos bsh, sistema operativo Hurd, aplicaciones, etc., bajo la licencia pública general GPL (General Public License), que permite usar, copiar, distribuir y modificar (con las mismas condiciones). Se conserva la firma del autor. Se puede cobrar.

Linux se crea con esta filosofía de libre distribución y el sistema operativo completo que se construye con este núcleo también. A todo el sistema se le da el nombre de GNU/Linux (distribución completa del sistema operativo con Linux), que contiene el núcleo (1,5 millones de líneas de código) más las otras capas del sistema operativo y utilidades. Si bien muchas veces se denomina a todo el sistema simplemente LINUX.

## 1.3. WINDOWS.

En 1981 IBM lanzó el PC de IBM basado en el procesador Intel 8088. El PC venía equipado con un sistema operativo en modo real de 16 bit monousuario, manejado por línea de comandos, llamado MS-DOS 1.0. Este sistema operativo había sido proporcionado por Microsoft, una compañía naciente, conocida por su intérprete de BASIC para sistemas 8080 y Z-80. Dos años más tarde salió a la luz un sistema operativo mucho más potente, llamado MS-DOS 2.0. Este sistema incluía un procesador de línea de comandos (shell) con varias características tomadas de UNIX. Con el paso de los años MS-DOS siguió adquiriendo nuevas funciones pero seguía siendo un sistema orientado a la línea de comandos.

Inspirado en la interfaz de usuario de la Apple Lisa (la precursora de la Apple Macintosh), Microsoft decidió dar a MS-DOS una interfaz gráfica, a la que denominó Windows. Windows 1.0 salió al mercado en 1985 pero no fue hasta 1990 con la versión 3.0 y siguientes que Microsoft alcanzó un gran éxito comercial. Ninguna de estas versiones cercanas era un sistema operativo sino más bien una interfaz gráfica de usuario colocada encima de MS-DOS. Todos los programas se ejecutaban en el mismo espacio de direcciones y un error de programación en cualquiera de ellos podía paralizar el sistema.

Windows 95 salió al mercado en 1995, no eliminó por completo MS-DOS aunque transfirió casi todas sus funciones a la parte Windows. Juntos Windows 95 y MS-DOS 7.0 contenían casi todas las funciones de un sistema operativo completo, incluidas memoria virtual, administración de procesos y multiprogramación.

Incluso en Windows 98 que salió al mercado en 1998, seguía presente el MS-DOS 7.1 y ejecutando código de 16 bits. Aunque había migrado más funcionalidad a la parte Windows y usaba de forma estándar una nueva organización de discos FAT32, en su interior no era muy diferente de Windows 95. El cambio sustancial residía en la más estrecha integración del escritorio e Internet.

En el año 2000 (año del milenio) Microsoft sacó una modernización menor de Windows 98 denominada Windows ME (Windows Millenium). Aunque se corregían algunos errores y se añadían unas cuantas funciones, en los aspectos internos era en esencia igual a Windows 98. Una nueva función que resultó interesante fue la capacidad de restaurar el ordenador a sus parámetros anteriores después de una configuración errónea.

## Windows NT.

A finales de los años ochenta, Microsoft decidió producir un sistema operativo de 32 bits completamente nuevo y compatible con Windows. Este sistema, llamado Windows NT (New Technologies) estaba pensado para aplicaciones de negocios «de misión crítica» y para usuarios caseros. En 1993 sale la primera versión con la denominación Windows NT 3.1. La reticencia de los usuarios a cambiarse al nuevo entorno, de mayores requerimientos de máquina y menos software disponible, llevó a Microsoft a elaborar Windows 95 y después Windows 98, indicando siempre que ésa sería la última versión de un sistema basado en MS-DOS.

La primera modernización importante llegó de la mano de Windows NT 4.0, con un interfaz semejante a Windows 95, en 1996, lo que facilitó la migración desde Windows 95 a este nuevo sistema.

Desde el principio NT se diseñó para que fuera portable. Con Windows NT 4.0 había versiones para Pentium, Alpha, MIPS y PowerPC, entre otras CPUs. A continuación se muestra una tabla resumen de diferencias entre Windows 95/98 y Windows NT:

Elemento	Windows 95/98	Windows NT
¿Sistema de 32 bits en su totalidad? .....	No	Sí
¿Seguridad? .....	No	Sí
¿Correspondencia de archivos protegidos? .....	No	Sí

.../...

.../...

¿Espacio de direcc. privado para cada prog. MS-DOS? .....	No	Sí
¿Unicode? .....	No	Sí
Ejecuta en .....	Intel 80x86	80x86, Alpha...
¿Manejo de multiprocesadores? .....	No	Sí
¿Código reentrante dentro del SO? .....	No	Sí
¿Plug and Play? .....	Sí	No
¿Administración de Energía? .....	Sí	No
¿Sistemas de archivos FAT-32? .....	Sí	Opcional
¿Sistemas de archivos NTFS? .....	No	Sí
¿API Win32? .....	Sí	Sí
¿Ejecuta todos los programas MS-DOS antiguos? .....	Sí	No
¿Los usuarios pueden escribir datos cruciales del SO? .....	Sí	No

## Windows 2000.

En 1999, Microsoft cambió el nombre de la siguiente versión de Windows NT 5.0 a Windows 2000, buscando presentar un nombre neutral que todos los usuarios, tanto de Windows NT como de Windows 98 vieran como el siguiente paso lógico en la evolución de ambos sistemas.

Windows 2000 representa un importante salto cualitativo en la familia de sistemas operativos de Microsoft, aunando en él las características de las dos ramas de sistemas operativos existentes hasta la fecha y deshaciéndose definitivamente de la carga del código de 16 bits, remanente en la familia 95/98/Me.

## Windows XP.

Liberado en el año 2001, en su versión *Home* y *Professional*, representa nuevamente una separación, que se creía cerrada, entre los sistemas operativos de servidor y los de usuario final.

Windows XP emplea el núcleo de Windows 2000, incorporando una nueva interfaz gráfica, haciendo alarde de mayores capacidades multimedia, de seguridad, etc.

Hoy en día, Windows XP es el sistema operativo de referencia de Microsoft para equipos de usuario final, tanto domésticos como profesionales.

## Windows 2003.

Acosado por las quejas sobre la falta de seguridad y estabilidad de sus sistemas operativos, Microsoft decidió dar un giro en su filosofía sobre la seguridad de sus sistemas operativos. Hasta este momento, los sistemas se instalaban por defecto «abiertos», esto es con todas sus funcionalidades acti-

vadas. Esta filosofía, provocaba graves agujeros de seguridad, sobre todo en sistemas mal administrados o mantenidos por personal inexperto.

El nuevo cambio de filosofía ha conducido a crear instalaciones «cerradas» por defecto, obligando al usuario o administrador de las mismas a conocer y configurar el sistema activando los servicios que realmente necesite.

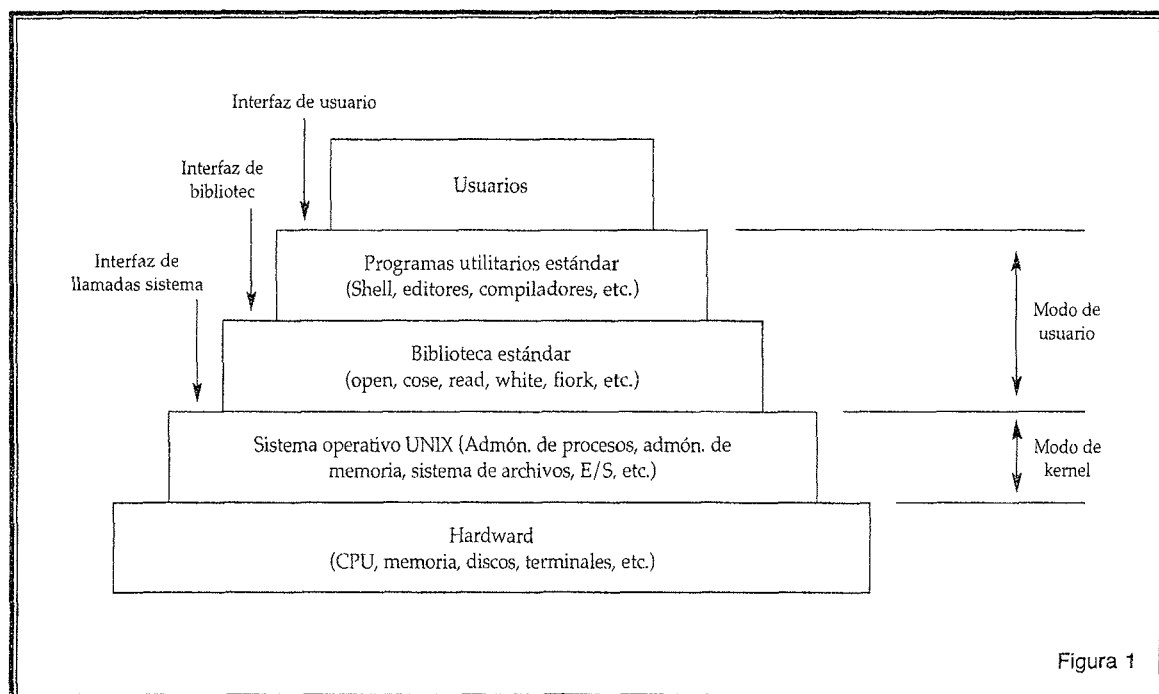
Como resultado del cambio, Microsoft publicó su nuevo sistema operativo Windows 2003, en abril del 2003, como una evolución más segura y estable de Windows 2000, aprovechando para incorporar las nuevas funcionalidades que el mercado demandaba, como el incipiente soporte para las arquitecturas de 64 bits.

Este cambio de filosofía también se ha reflejado en Windows XP, si bien en este caso en forma de un «Paquete de actualización» o Service Pack, conocido como el Service Pack 2 que todavía plantea demasiadas dudas y recelos en los administradores, sobre la conveniencia o no de su implantación, pues el extremado incremento de seguridad, imposibilita en muchos casos el correcto funcionamiento de aplicaciones y servicios necesarios.

## 2. UNIX, LINUX.

### 2.1. UNIX.

El núcleo de UNIX (kernel) se clasifica como de tipo monolítico, pero en él se pueden encontrar dos partes principales: el núcleo dependiente de la máquina y el núcleo independiente. El núcleo dependiente se encarga de las interrupciones, los manejadores de dispositivos de bajo nivel (lower half) y parte del manejo de la memoria. El núcleo independiente es igual en todas las plataformas e incluye el manejo de llamadas del sistema, la planificación de procesos, el entubamiento, el manejo de señales, la paginación e intercambio, el manejo de discos y del sistema de archivos.





Las ideas principales de UNIX fueron derivadas del proyecto MULTICS (Multiplexed Information and Computing Service) del MIT y de General Electric. Estas ideas son:

- Todo se maneja como cadena de bytes: los dispositivos periféricos, los archivos y los comandos pueden verse como secuencias de bytes o como entes que las producen. Por ejemplo, para usar una terminal en UNIX se hace a través de un archivo (generalmente en el directorio /dev y con nombre ttyX).
- Manejo de tres descriptores estándares: todo comando posee tres descriptores por omisión llamados «stdin», «stdout» y «stderr», los cuales son los lugares de donde se leen los datos de trabajo, donde se envían los resultados y en donde se envían los errores, respectivamente. El 'stdin' es el teclado, el «stdout» y el «stderr» son la pantalla por omisión (default).
- Capacidades de «entubar» y «redireccionar»: El «stdin», «stdout» y el «stderr» pueden usarse para cambiar el lugar de donde se leen los datos, donde se envían los resultados y donde se envían los errores, respectivamente. A nivel de comandos, el símbolo de «mayor que» (>) sirve para enviar los resultados de un comando a un archivo, mientras que el símbolo (|) conocido como pipe o entubamiento, redirige la salida de un comando a la entrada estándar del comando siguiente (en la misma línea de comando). Por ejemplo, en UNIX el comando «ls» lista los archivos del directorio actual (es lo mismo que «dir» en DOS). Si en lugar de ver los nombres de archivos en la pantalla se quieren guardar en el archivo «listado», el redireccionamiento es útil y el comando para hacer la tarea anterior es «ls > listado». Si lo que se desea es enviar a imprimir esos nombres, el «entubamiento» es útil y el comando sería «ls | lpr», donde el símbolo «|» (pipe) es el entubamiento y «lpr» es el comando para imprimir en UNIX BSD.
- Crear sistemas grandes a partir de módulos: cada instrucción en UNIX está diseñada para poderse usar con «pipes» o «redireccionamiento», de manera que se pueden crear sistemas complejos a través del uso de comandos simples y elegantes. Un ejemplo sencillo de esto es el siguiente. Suponga que se tienen cuatro comandos separados A, B, C y D cuyas funcionalidades son:

A: lee matrices comprobando tipos de datos y formato.

B: recibe matrices, las invierte y arroja el resultado en forma matricial.

C: recibe una matriz y le pone encabezados «bonitos».

D: manda a la impresora una matriz cuidando el salto de página, etc.

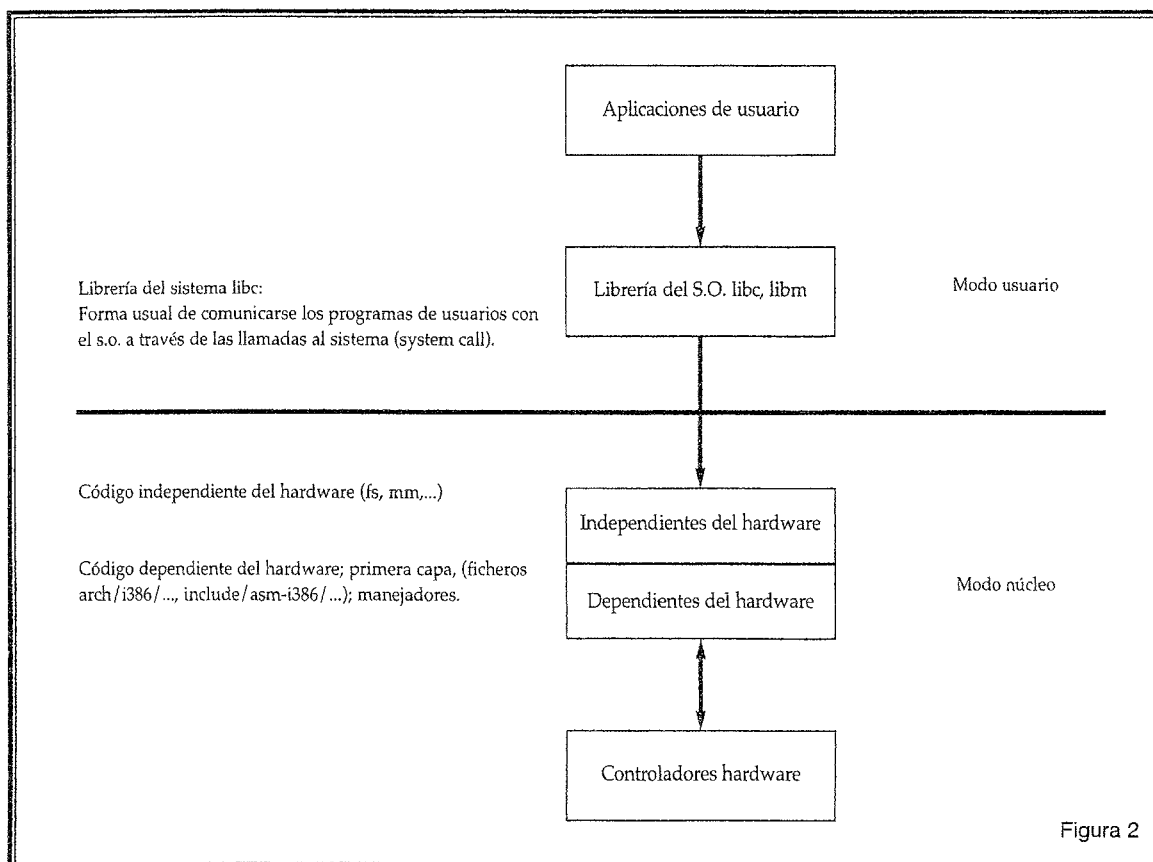
Como se ve, cada módulo hace una actividad específica, si lo que se quiere es un pequeño sistema que lea un sistema de ecuaciones y como resultado se tenga un listado «bonito», simplemente se usa el entubamiento para leer con el módulo A la matriz, que su resultado lo reciba el B para obtener la solución, luego esa solución la reciba el módulo C para que le ponga los encabezados «bonitos» y finalmente eso lo tome el módulo D y lo imprima, el comando completo sería «A | B | C | D». ¿Fácil no?

## 2.2. LINUX.

### 2.2.1. Características del núcleo.

- Su código es de libre uso.
- Escrito en lenguaje C, compilado con gcc, primera capa en ensamblador.

- Ejecutable en varias plataformas hardware.
- Se ejecuta en máquinas con arquitectura de 32 bits y 64 bits.
- Estaciones de trabajo y servidores.
- Código y funcionamiento escrito bajo la familia de estándares POSIX (Portable Operating System Interface).
- Soporta CPU's con uno o varios microprocesadores (SMP) symmetric multiprocessing.
- Multitarea.
- Multiusuario.
- Gestión y protección de memoria.
- Memoria virtual.
- Varios sistemas de ficheros.
- Comunicación entre procesos (señales, pipes, IPC, sockets).
- Librerías compartidas y dinámicas.
- Permite trabajar en red TCP/IP.
- Soporte gráfico para interfase con el usuario.



- Estable, veloz, completo y rendimiento aceptable.
- Funcionalmente es muy parecido a UNIX.
- Mas de 100.000 usuarios.
- Actualizado, mejorado, mantenido y ampliado por la comunidad de usuarios (modelo «bazar», contrapuesto al modelo «catedral»).

Linux mantiene una estructura monolítica, pero admite módulos cargables.

### 2.3. PROCESOS Y THREADS.

El manejo de procesos en UNIX es por prioridad y round robin. En algunas versiones se maneja también un ajuste dinámico de la prioridad de acuerdo al tiempo que los procesos han esperado y al tiempo que ya han usado el CPU. El sistema provee facilidades para contabilizar el uso de CPU por proceso y una pila común para todos los procesos cuando necesitan estarse ejecutando en modo privilegiado (cuando hicieron una llamada al sistema). UNIX permite que un proceso haga una copia de sí mismo por medio de la llamada «fork», lo cual es muy útil cuando se realizan trabajos paralelos o concurrentes; también se proveen facilidades para el envío de mensajes entre procesos (pipes, signals).

Los procesos no interactivos se denominan daemons o procesos background. Cuando se inicia un proceso, se le asigna un identificador PID, se guarda el proceso que lo lanzó PPID, el propietario que lo lanzó UID y el grupo de pertenencia GID, lo que definirá el perfil de permisos de acceso a los que tendrá derecho. Existe la posibilidad de alterar el usuario o grupo efectivo de permisos, durante la ejecución del proceso, mediante la llamada a `setuid` o `setgid`, siempre que se disponga de los permisos apropiados. También existe una bandera de permisos `setuid` asociada al archivo del programa que permite ejecutar éste, con los permisos del propietario del archivo en lugar de los del usuario que lo ejecuta.

LINUX combina multiprogramación y tiempo compartido. Es un sistema preemptive, con planificación dependiente de la categoría del proceso: tiempo real o proceso ordinario. Sobre procesos ordinarios emplea la técnica de planificación por prioridad dinámica (la asignada inicialmente más una variable asignada por el sistema). En tiempo real se aplica un planificador FIFO non-preemptive para ciertas tareas de igual prioridad o Round Robin preemptive en otros casos, también de tiempo real.

### 2.4. GESTIÓN DE MEMORIA.

Los primeros sistemas con UNIX nacieron en máquinas cuyo espacio de direcciones era muy pequeño (por ejemplo 64 kilobytes) y tenían un manejo de memoria real algo complejo. Actualmente, todos los sistemas UNIX utilizan el manejo de memoria virtual siendo el esquema más usado la paginación por demanda y combinación de segmentos paginados, en ambos casos con páginas de tamaño fijo. En todos los sistemas UNIX se usa una partición de disco duro para el área de intercambio. Esa área se reserva al tiempo de instalación del sistema operativo. Una regla muy difundida entre administradores de sistemas es asignar una partición de disco duro que sea al menos el doble de la cantidad de memoria real del ordenador. Con esta regla se permite que se puedan intercambiar flexiblemente todos los procesos que estén en memoria RAM en un momento dado por otros que estén en el disco. Todos los procesos que forman parte del kernel no pueden ser intercambiados a disco. Algunos sistemas operativos (como SunOS) permiten incrementar el espacio de intercambio incluso mientras el sistema está en uso (en el caso de SunOS con el comando «swapon»). También es muy importante que al mo-

mento de decidirse por un sistema operativo se pregunte por esa facilidad de incrementar el espacio de intercambio, así como la facilidad de añadir módulos de memoria RAM a la computadora sin necesidad de reconfigurar el núcleo.

Cada proceso dispone de su propio espacio de direcciones, organizado en segmentos según: Text Segment, código; Data Segment, datos, este segmento tiene dos partes: Initialized Data -datos inicializados- y Uninitialized Data -datos no inicializados- conocido como BSS; Stack Segment.

Es posible compartir código entre procesos mediante el empleo de Shared Text Segments. Dos procesos nunca comparten los segmentos de datos y de pila (salvo los thread), la forma de compartir información se lleva a cabo mediante el empleo de segmentos especiales de memoria compartida Shared Segments.

Swapper (PID 0)

Init (PID 1)

Page Daemon (PID 2)

En 4BSD, la memoria principal consta de 3 partes: las dos primeras, el kernel y el mapa central (core map), están fijas en la memoria y nunca se pagan a disco; el resto de la memoria se divide en marcos de página, cada uno de los cuales puede contener una página de texto (código), datos o de pila, contener una página de la tabla de páginas o estar en la lista libre.

El mapa central incluye información acerca del contenido de los marcos de página, donde cada entrada describe el marco de página correspondiente (la 0 el marco 0 y así consecutivamente).

Reemplazo: cada 250ms el page daemon mira si el número de marcos libres no baja de cierta cantidad, (aprox 1/4 del tamaño de la memoria). Para la paginación se emplea un algoritmo conocido como algoritmo de reloj de una manecilla, examinando de forma circular los marcos de página. Es un algoritmo global. 4BSD emplea una versión modificada con 2 «manecillas», mientras que System V emplea el algoritmo de una manecilla, colocando los marcos en la lista de marcos libres, si no se usa durante n pasadas consecutivas.

Si hay reemplazo el swapper desaloja uno o varios procesos que hayan estado inactivos durante más de 20s, o el que haya estado más tiempo en memoria entre los cuatro más grandes.

En LINUX, cada proceso obtiene 3GB de espacio virtual de direcciones (en máquinas de 32 bits), reservándose el cuarto giga para sus tablas de páginas y otros datos del kernel. El espacio de direcciones virtual se divide en áreas o regiones homogéneas y contiguas, alineadas a fronteras de página. Cada área consiste en una serie de páginas consecutivas con las mismas propiedades en cuanto a protección y paginación. LINUX emplea tablas de páginas de tres niveles. Es un sistema de paginación por demanda sin paginación anticipada.

dir virtual: (p1 p2 p3 off)

## 2.5. ENTRADA/SALIDA.

Derivado de la filosofía de manejar todo como flujo de bytes, los dispositivos son considerados como archivos que se acceden mediante descriptores de archivos cuyos nombres se encuentran gene-

ralmente en el directorio «/dev». Cada proceso en UNIX mantiene una tabla de archivos abiertos (donde el archivo puede ser cualquier dispositivo de entrada/salida). Esa tabla tiene entradas que corresponden a los descriptores, los cuales son números enteros obtenidos por medio de la llamada del sistema «open». En la tabla 2 se muestran las llamadas más usuales para realizar entrada/salida.

TABLA 2

**LLAMADAS AL SISTEMA DE ENTRADA/SALIDA**

LLAMADA	FUNCIÓN
open	Obtener un descriptor entero.
close	Terminar las operaciones sobre el archivo.
lseek	Posicionar la entrada/salida.
read,write	Leer o escribir al archivo (dispositivo).
ioctl	Establecer el modo de trabajo del dispositivo.

En UNIX es posible ejecutar llamadas al sistema de entrada/salida de dos formas: síncrona y asíncrona. El modo síncrono es el modo normal de trabajo y consiste en hacer peticiones de lectura o escritura que hacen que el originador tenga que esperar a que el sistema le responda, es decir, que le de los datos deseados. A veces se requiere que un mismo proceso sea capaz de supervisar el estado de varios dispositivos y tomar ciertas decisiones dependiendo de si existen datos o no. En este caso se requiere una forma de trabajo asíncrona. Para este tipo de situaciones existen las llamadas a las rutinas «select» y «poll» que permiten saber el estado de un conjunto de descriptores.

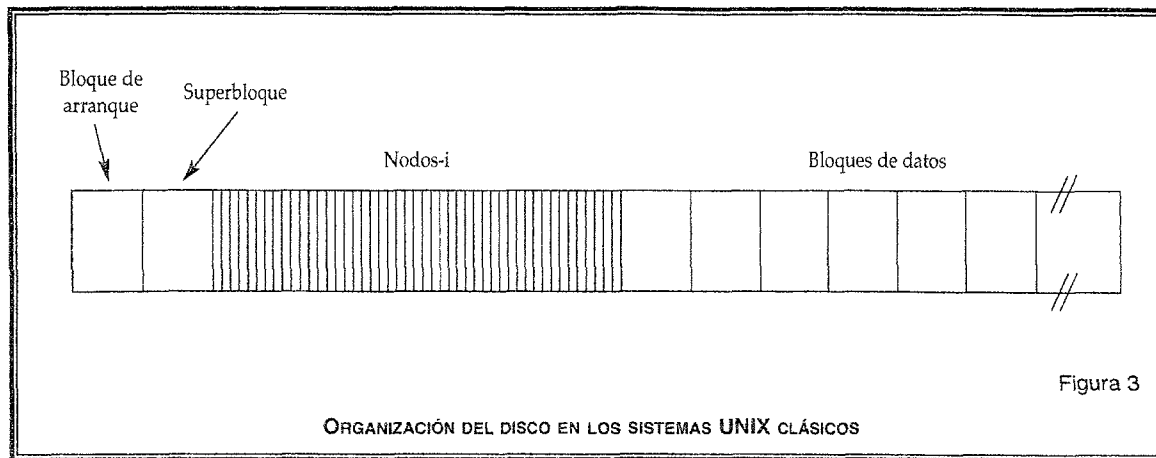
Hay dos tipos de dispositivos: de bloque (discos, cintas...) y de carácter (terminales, impresoras...).

Para la E/S de red se emplean los SOCKETS (berkeley) que una vez abierto se trata como un fichero.

## 2.6. EL SISTEMA DE ARCHIVOS.

El sistema de archivos de UNIX, que hoy identificamos como el sistema tradicional, tuvo su aparición con la versión 7 y fue empleado sobre la PDP-11. Posteriormente, la distribución de Berkeley introdujo mejoras significativas. También se usan otros sistemas de archivos. Todos los sistemas UNIX pueden manejar múltiples particiones de disco, cada una con un sistema de archivos distinto.

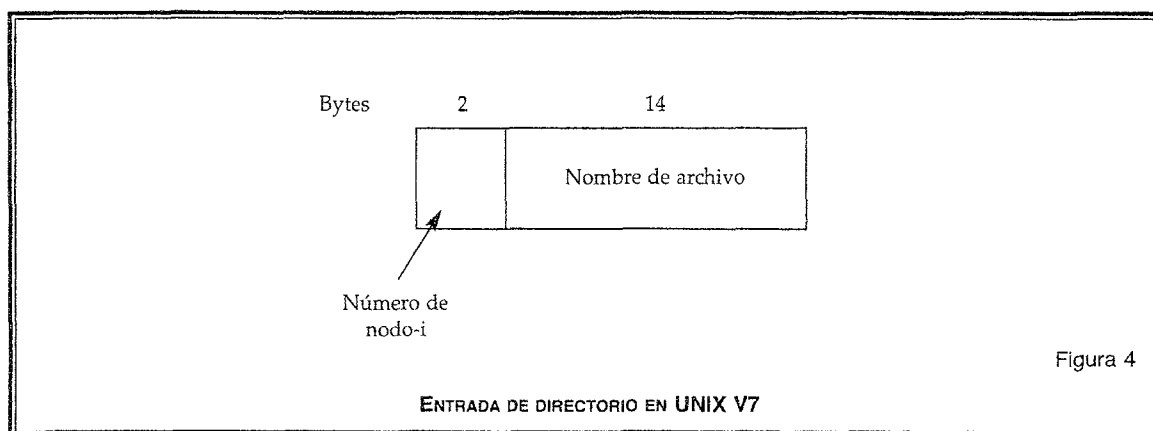
Una partición de disco clásica en UNIX contiene un sistema de archivos con la organización que se ilustra en la figura siguiente. UNIX no usa el bloque 0, que a menudo contiene código para arrancar el ordenador. El bloque 1 es el superbloque; contiene información crucial acerca de la organización del sistema de archivos, incluido el número de nodos-i (i-nodes), el número de bloques de disco y el principio de la lista de bloques de disco libres. La destrucción del superbloque hace que el sistema de archivos ya no pueda leerse.



Después del superbloque vienen los nodos-i (abreviatura de nodos-índice, aunque nunca se les llama así), que están numerados del uno hasta algún máximo. Cada nodo-i tiene 64 bytes de largo y describe exactamente un archivo. Un nodo-i contiene información de contabilidad (incluida toda la información devuelta por stat, que tan sólo la toma del nodo-i), así como suficiente información para localizar todos los bloques de disco que contienen los datos del archivo.

Después de los nodos-i están los bloques de datos. Aquí se almacenan todos los archivos y directorios. Si un archivo o directorio consta de más de un bloque, los bloques no necesitan estar contiguos en el disco. De hecho, lo más probable es que los bloques de un archivo grande estén dispersos por todo el disco. Es precisamente esta dispersión lo que se buscó reducir con las mejoras de Berkeley.

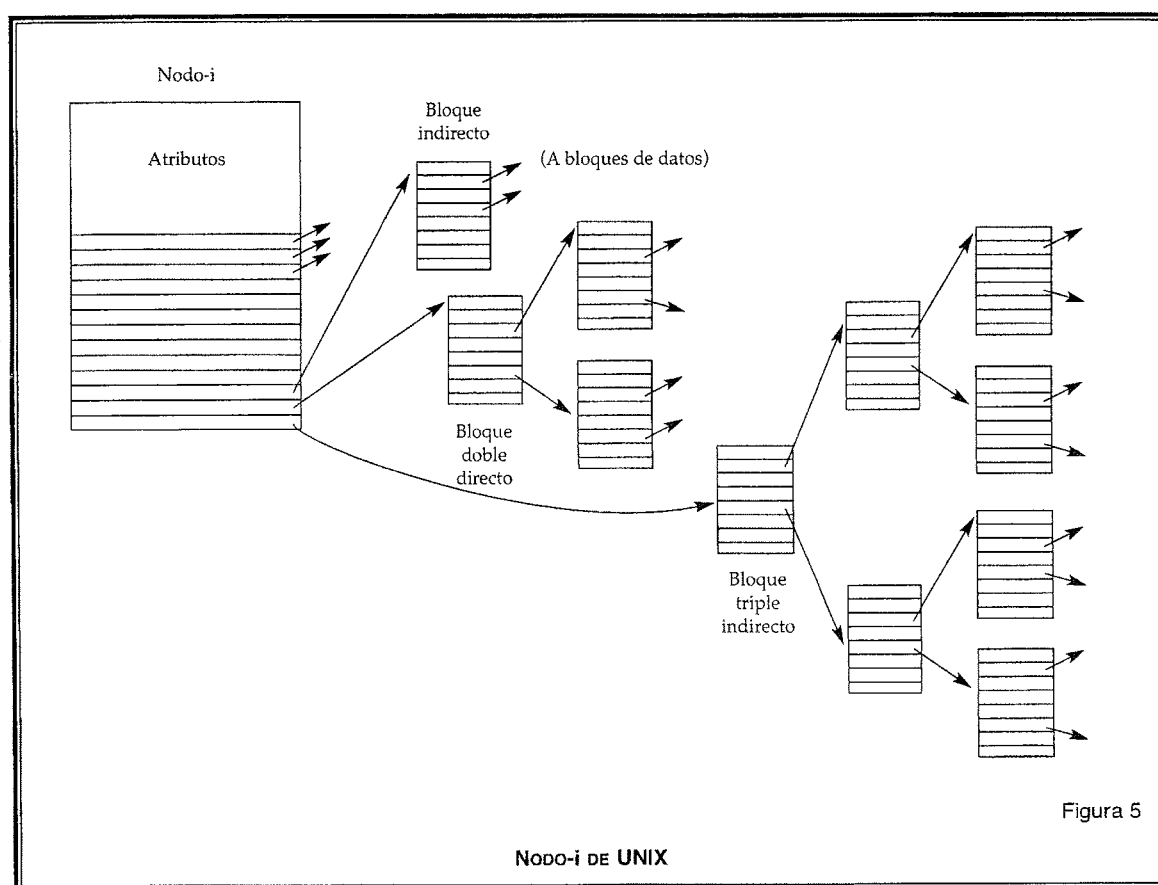
Un directorio en el sistema de archivos tradicional consiste en una colección no ordenada de entradas de 16 bytes. Cada entrada contiene un nombre de archivo (hasta 14 caracteres arbitrarios) y el número de nodo-i del archivo, como se muestra en la figura siguiente. Para abrir el archivo en el directorio de trabajo, el sistema tan sólo lee el directorio y compara el nombre buscado con cada entrada hasta encontrar el nombre, o bien concluye que no está presente. Dado que cada archivo del sistema de archivos necesita de un nodo-i y que este se referencia desde la entrada de directorio con 2 bytes, el número máximo de archivos por sistema de archivos es de 64K.



Si el archivo está presente, el sistema extrae el número de nodo-i y lo usa como índice para consultar la tabla de nodos-i (en disco), localizar el nodo-i correspondiente y traerlo a la memoria. El no-

do-i se coloca en la tabla de nodos-i, una estructura de datos del kernel que contiene los nodos-i de todos los archivos y directorios que están abiertos en ese momento.

Una vez localizado el nodo-i del archivo deseado, éste contiene, además de la información de seguridad y contabilidad necesarias, las direcciones en disco de los primeros 10 bloques del archivo. En caso de archivos de más de 10 bloques, un campo en el nodo-i contiene la dirección en disco de un bloque de indirección sencilla, este bloque (como se muestra en la figura siguiente) contiene las direcciones en disco de más bloques del archivo. Si no es suficiente, el siguiente campo en el nodo-i contiene la dirección en disco de un bloque de doble indirección, que contiene las direcciones en disco de más bloques de indirección sencilla, que a su vez contienen la dirección en disco de más bloques del archivo. Por último, si aún es preciso un archivo mayor, el último campo del nodo-i contiene la dirección de un bloque de triple indirección que, con la misma lógica de los casos anteriores, contiene las direcciones en disco de más bloques de doble indirección.



Suponiendo un tamaño de bloque de 1KB, y una dirección de disco de 4 bytes, tendremos los siguientes valores:

10 direcciones bloque  $\times$  1KB = 10KB.

1 bloque indirecto =  $1024/4$  direcciones = 256 direcciones  $\times$  1KB = 256 KB.

1 bloque doble indirecto = 256 bloques indirectos  $\times$  256 KB/bloque indirecto = 65.536 KB.

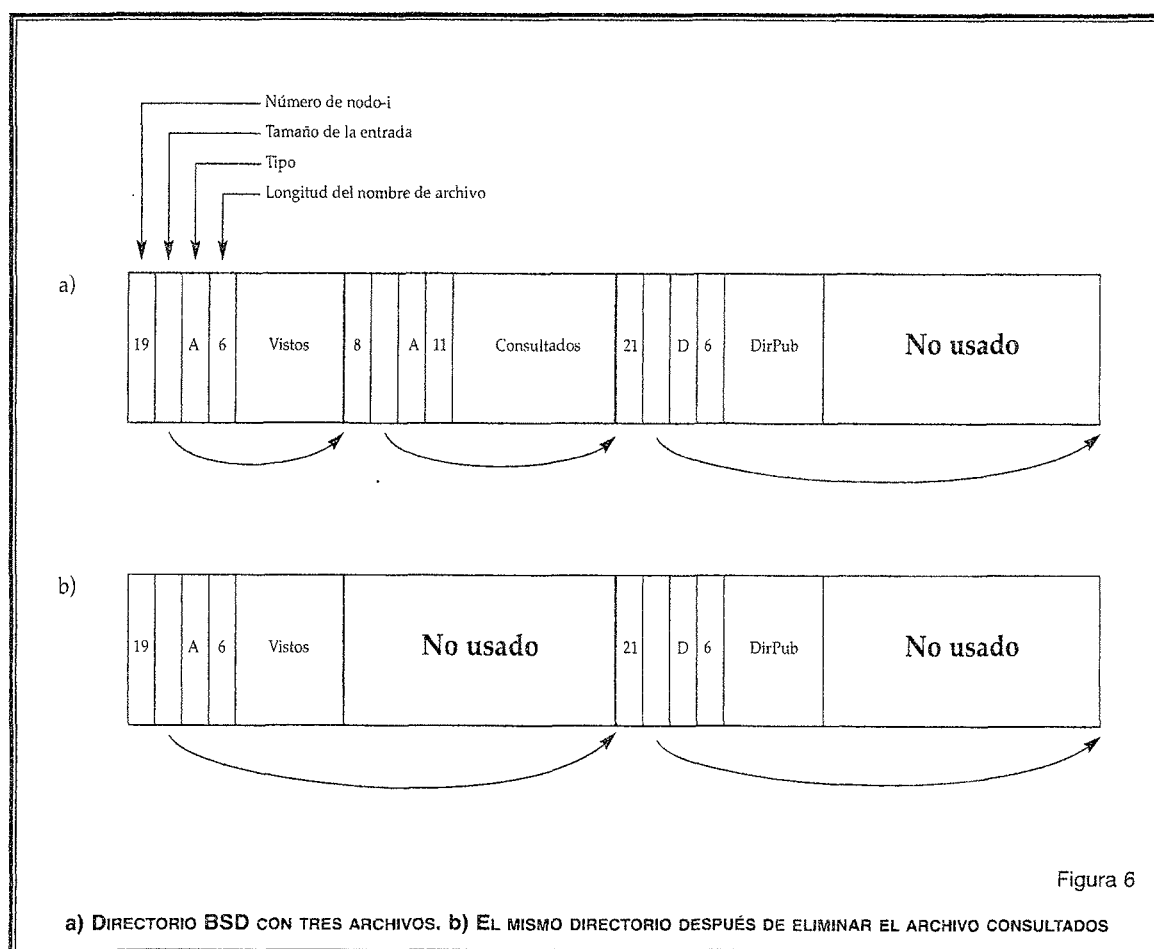
1 bloque triple indirecto = 256 bloques dobles  $\times$  65.536 KB/bloque doble = 16.777.216 KB.

Lo que supone un límite total para el tamaño de archivo de 16,06 GB.

Además del sistema de archivos clásico de UNIX, existen otros sistemas de archivos que pueden emplearse en UNIX y LINUX. Berkeley mejoró el sistema clásico con el BFFS (Berkeley Fast File System), que fue quien introdujo los nombres de archivo de 255 caracteres (el clásico solamente admitía 14). Berkeley tuvo que introducir cuatro llamadas al sistema nuevas *opendir*, *closedir*, *readdir* y *rewinddir* para que los programas pudieran leer los directorios sin conocer su estructura interna. Posteriormente estos cambios se añadieron a POSIX y a todas las versiones de UNIX.

La mejora de Berkeley al sistema de archivos clásico de UNIX, se concretó en tres cambios importantes: En primer lugar, se reorganizaron los directorios. En la nueva estructura, reseñada en la figura siguiente, cada directorio consta de algún número entero de bloques de disco, lo que permite escribir de forma atómica los directorios en el disco. Dentro de un directorio, las entradas de archivos y directorios aparecen en desorden, cada una inmediatamente después de la que la precede. Una entrada no puede cruzar una frontera de bloque, así que a menudo hay cierto número de bytes desocupados (de relleno) al final de cada bloque de disco.

Cada entrada de directorio consta de cuatro campos de longitud fija y uno de longitud variable. El primer campo es el número de nodo-i, a continuación viene un campo que contiene el tamaño total de la propia entrada, incluyendo el posible relleno después del nombre de archivo. Este campo permite localizar la siguiente entrada de directorio. A continuación viene el campo de tipo (archivo, directorio, etc.). El último campo fijo es la longitud del nombre de archivo en bytes. Al final se encuentra el nombre de archivo terminado con un byte 0 (el carácter NULL) y rellenado hasta una alineación de 32 bits.





En la segunda parte de la figura, puede observarse el estado del directorio después de haber eliminado una entrada. El campo tamaño de la entrada precedente a la borrada se actualiza, incorporando como relleno el espacio ocupado por la entrada eliminada, apuntando correctamente a la que es su próxima entrada en el directorio.

El segundo cambio importante que introdujo Berkeley en el sistema de archivos fue la división del disco en grupos de cilindros, cada uno con su propio superbloque, nodos-i y bloques de datos. Lo que se buscó con este cambio fue mantener cercanos nodo-i y los bloques de datos de un archivo dado, para evitar desplazamientos largos del brazo del disco. Siempre que sea posible, los bloques de disco se asignan en el grupo de cilindros que contiene el nodo-i.

El tercer cambio fue la introducción de dos tamaños de bloque en vez de sólo uno. Si se van a almacenar archivos grandes es más eficiente tener un número reducido de bloques grandes, en vez de muchos bloques pequeños. Por otra parte, muchos archivos UNIX son pequeños, por lo que tener sólo bloques grandes implicaría un desperdicio de espacio de disco. Tener dos tamaños de bloque permite efectuar transferencias eficientes de archivos grandes y aprovechar de forma eficiente el espacio en el caso de archivos pequeños. El precio a pagar es un aumento considerable en la complejidad del código.

LINUX comenzó empleando el sistema de archivos de MINIX, pero estaba limitado a nombres de 14 caracteres y a archivos de 64 MB de tamaño. La primera mejora vino de la mano de un sistema de archivos denominado Ext que permitía nombres de archivo de 255 caracteres y 2 GB de tamaño por archivo, pero era muy lento. La evolución vino de la mano del sistema de archivos Ext2, con nombres de archivo largos, archivos grandes y mejor rendimiento, siendo en la actualidad el sistema de archivos por defecto en LINUX. Ext2 es muy similar a BFFS, con algunas diferencias menores.

El sistema de archivos de UNIX ofrece un poderoso conjunto de comandos y llamadas al sistema. En la tabla 3 se muestran los comandos más útiles para el manejo de archivos en UNIX.

TABLA 3

### COMANDOS UNIX DEL SISTEMA DE ARCHIVOS

UNIX	UTILIDAD
rm	Borra archivos
cp	Copia archivos
mv	Renombra archivos
ls	Lista el contenido de un directorio
mkdir	Crea un directorio
rmdir	Borra directorios
ln	Crea enlaces (entradas a un mismo archivo físico desde ubicaciones diferentes), tanto enlaces «físicos» (hard-links) como «simbólicos» (soft-links). Un enlace simbólico guarda la ruta del archivo referenciado, puede estable-

.../...

.../...

	cerse entre particiones distintas y no desaparece si se elimina el archivo a que hace referencia. Un enlace físico solamente se puede establecer en directorios de la misma partición, para eliminar un archivo se deben eliminar todos los enlaces físicos que lo referencien. Todos los enlaces físicos de un archivo comparten el propietario y los permisos de acceso
chmod	Establece y modifica los permisos de acceso a archivos y directorios
chown	Cambia el propietario de un archivo o directorio. Se puede emplear por el propietario actual o por el superusuario
chgrp	Cambia el grupo de pertenencia de un archivo o directorio. Se puede emplear por el propietario o por el superusuario

Con objeto de compartir los archivos entre ordenadores conectados en una red, a menudo con diferentes sistemas de archivos, Sun Microsystems desarrolló el sistema de archivos de red NFS (Network File System), que se usa en todos los sistemas UNIX modernos, incluido LINUX.

NFS se basa en la idea de permitir que una colección arbitraria de clientes y servidores, compartan un mismo sistema de archivos. NFS puede ejecutarse entre ordenadores situados en redes distintas e incluso en un solo ordenador, que ejecuta las funciones de servidor y cliente al mismo tiempo. NFS implementa sus propios protocolos de comunicación entre los clientes y los servidores. En resumen, los equipos servidores exportan una lista de subdirectorios a compartir, lo que incluye toda la jerarquía que tienen por debajo y los clientes conectan esos recursos a sus propias jerarquías haciendo el acceso completamente homogéneo y transparente al usuario.

## 2.7. SEGURIDAD.

La seguridad en el entorno UNIX, se basa en la existencia de usuarios y grupos registrados. A cada usuario se le asigna un identificador uid (User Identification) y un gid (Group Identification) del grupo de pertenencia, en el archivo /etc/passwd los uid de valor inferior a 100 tienen unos privilegios especiales, siendo el usuario con uid 0 el conocido como root o superusuario del sistema.

La seguridad de acceso de los sistemas UNIX, reside en tres archivos de texto ubicados en el directorio /etc:

/etc/passwd	Archivo de registro de usuarios. Cada usuario está identificado en una línea de texto que contiene: Nombre de Usuario, uid, gid, descripción, directorio base, shell a ejecutar. Es un archivo de lectura para todo el mundo.
/etc/shadow	Archivo de registro de contraseñas. Cada línea del archivo passwd tiene su correspondencia en este archivo. Para cada usuario registrado se almacena la contraseña encriptada y los parámetros de validez y duración máxima y mínima de la misma. Solamente el administrador tiene acceso a leer este archivo.
/etc/group	Archivo de registro de grupos. Cada grupo diferente está identificado en una línea de texto que contiene: nombre del Grupo, gid, lista de usuarios que pertenecen al

grupo. Es un archivo de lectura para todo el mundo. Un usuario puede estar registrado en más de un grupo.

La protección de archivos en UNIX se maneja por medio de una cadena de permisos de nueve caracteres. Los nueve caracteres se dividen en tres grupos de tres caracteres cada uno.

rwX	rwX	rwX
1	2	3

El primer grupo (1) especifica los permisos del dueño del archivo owner. El segundo grupo (2) especifica los permisos para aquellos usuarios que pertenecen al mismo grupo de trabajo que el dueño, group, y finalmente el tercer grupo (3) indica los permisos para el resto del mundo others o world. En cada grupo de tres caracteres pueden aparecer las letras RWX en ese orden indicando permiso de leer (Read), escribir (Write) y ejecutar (eXecute). Por ejemplo, la cadena completa `rwxr-xr--` indica que el dueño tiene los tres permisos (Read, Write, Execute), los miembros de su grupo de trabajo tienen permisos de leer y ejecutar (Read, Execute) y el resto del mundo sólo tienen permiso de leer (Read). Las llamadas al sistema más útiles en UNIX son «open», «close» e «ioctl». Sirven para abrir, cerrar archivos; y establecer las características de trabajo. Por ejemplo, ya que en UNIX las terminales se acceden a través de archivos especiales, el «ioctl» (input output control) sirve para establecer la velocidad, paridad, etc. de la terminal.

### 3. FAMILIA WINDOWS.

#### 3.1. VISIÓN GENERAL.

Los sistemas operativos de Microsoft para los PCs de escritorio y portátiles pueden dividirse en tres familias: MS-DOS, Windows para consumidor (Windows 95/98/Me) y Windows NT.

Una visión general de estas familias se ha visto al hablar de la historia de los sistemas Microsoft. Nos centraremos ahora en Windows 2000 como sistema que supone la concurrencia de ambas familias Windows y el abandono definitivo de los sistemas basados en MS-DOS.

Windows 2000 hereda muchas propiedades de Windows NT 4.0. Se trata de un verdadero sistema multitarea de 32 bits, con procesos protegidos de manera individual. Cada proceso tiene un espacio de direcciones virtuales de 32 bits, paginado por demanda. El sistema operativo se ejecuta en modo kernel, mientras que los procesos de usuario lo hacen en modo usuario, ofreciendo protección completa. Los procesos pueden tener uno o más subprocesos, que el sistema operativo puede ver y planificar. Se cuenta con seguridad C-2, del Departamento de Defensa americano, para todos los archivos, directorios, procesos y demás objetos que pueden compartirse (si se retira el disco flexible y se desconecta la red). El sistema tiene soporte completo para el multiproceso simétrico en hasta 32 procesadores.

Aunque es la evolución de Windows NT 4.0, incorpora funciones propias de Windows 98 como el soporte Plug and Play, bus USB, IEEE 1394 (FireWire), IrDA y administración de energía, entre otras cosas. Se han incluido además funciones no presentes en ningún sistema anterior de Microsoft como el Servicio de Directorio Activo, seguridad con Kerberos, manejo de tarjetas inteligentes y una infraestructura para la administración del sistema y objetos de trabajo. Además, se ha extendido el principal sistema de archivos NTFS, para manejar archivos cifrados, cuotas de ocupación, archivos

enlazados, volúmenes dinámicos y el almacén de instancia única, que funciona de manera que dos usuarios pueden compartir el mismo archivo enlazado hasta que uno de ellos escribe en él, momento en el cual se crea una copia de forma automática.

Otra mejora muy importante es la internacionalización. Se han sacado las cadenas de texto incorporadas al código y se han puesto en directorios distintos del sistema operativo. Windows 2000 cuenta con un solo binario que funciona en cualquier lugar del mundo, decidiéndose en el momento de la instalación el idioma que se utilizará. Windows 2000 utiliza internamente Unicode para poder manejar idiomas que no usan el alfabeto latino.

Windows 2000 no tiene MS-DOS. Dispone de una interfaz de línea de comandos que es una aplicación de 32 bits que incluye la funcionalidad del antiguo MS-DOS además de otras funciones.

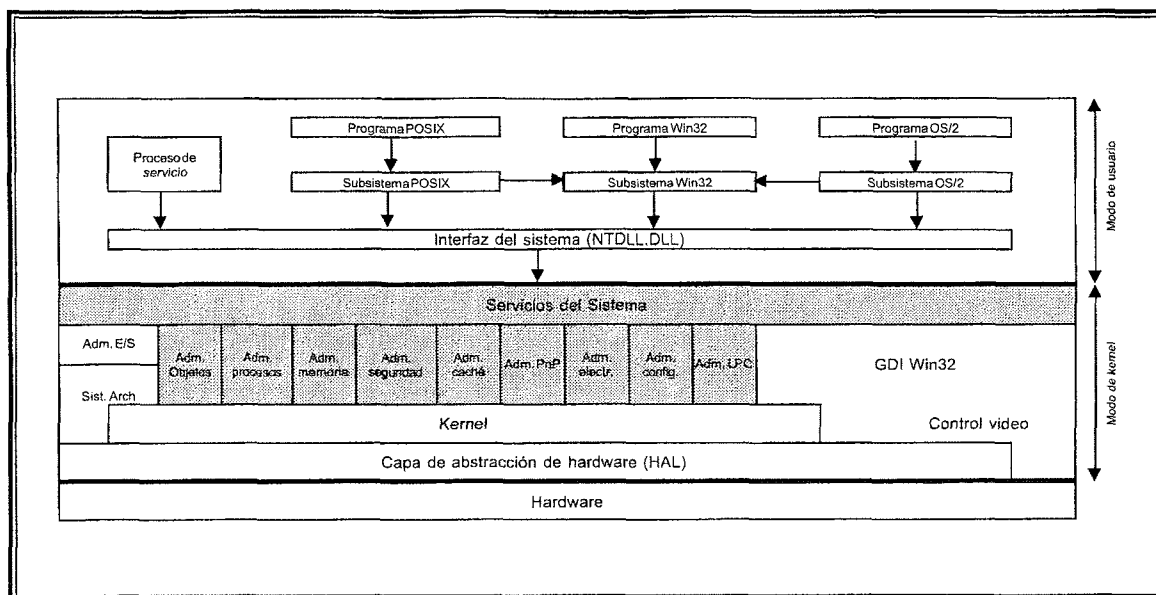
Windows 2000 ha disminuido su portabilidad pues ahora solamente está disponible para dos plataformas: Pentium e Intel IA-64, habiendo desechado las otras por motivos comerciales. También por motivos comerciales, existen diferentes niveles de producto, que por ahora son Professional, Server, Advanced Server y Datacenter Server, si bien el binario es el mismo, en el registro figura el nivel adquirido lo que limita ciertas características, que se detallan en la siguiente tabla:

Versión	RAM máx.	CPUs	Máx. clientes	Tam. cluster	Optimizado para
Professional	4 GB	2	10	0	Tiempo de respuesta
Server	4 GB	4	ilimitados	0	Vel. real de transporte
Advanced Server	8 GB	8	ilimitados	2	Vel. real de transporte
Datacenter Server	64 GB	32	ilimitados	4	Vel. real de transporte

Una última diferencia es que los servidores incluyen algo de software adicional y la versión Datacenter cuenta con herramientas adicionales para administrar trabajos grandes.

Windows 2000 cuenta, como el resto de los sistemas operativos, con un conjunto de llamadas al sistema que puede ejecutar. Sin embargo, Microsoft nunca ha publicado la lista de llamadas, y además las cambia de una versión a la siguiente. Lo que ha hecho es definir un conjunto de llamadas a funciones denominado interfaz de programación de aplicaciones (API) Win32, que se conoce de manera pública y está plenamente documentado. Se trata de procedimientos de biblioteca que efectúan llamadas al sistema para realizar el trabajo o, en algunos casos, lo realizan ellos mismos. Las llamadas de la API Win32 existentes no cambian al cambiar de versión, si bien se suelen añadir algunas nuevas.

Windows 2000 consta de dos componentes principales: el sistema operativo en sí, que se ejecuta en modo kernel, y los subsistemas de entorno, que se ejecutan en modo de usuario.



Mientras que el kernel es tradicional, en el sentido de que se encarga de la administración de procesos, la administración de memoria, los sistemas de archivos, etc. Los subsistemas de entorno son procesos individuales que ayudan a los programas de usuario a realizar ciertas funciones del sistema.

Si bien las primeras versiones de Windows NT siguieron una arquitectura de microkernel, donde los procesos de usuario interactuaban con los procesos de servidor empleando un modelo cliente-servidor, por razones de rendimiento a partir de Windows NT 4.0, casi todo el sistema operativo volvió a ponerse en modo de kernel, diseño que se ha perpetuado en Windows 2000. Desde la versión 4.0 de Windows NT ni ésta ni Windows 2000 se pueden considerar arquitecturas microkernel. Windows 2000 mantiene una cierta estructura interna, donde el sistema está dividido en varias capas, cada una de las cuales usa los servicios de las que están abajo.

En la parte superior del sistema operativo, por encima del denominado kernel y de los controladores de dispositivos (pero ejecutándose en modo kernel) se halla el ejecutivo (executive). Consta de 10 componentes, cada uno de los cuales es un conjunto de procedimientos que colaboran para alcanzar una meta.

Existen tres tipos de componentes de modo de usuario: DLLs, subsistemas de entorno y procesos de servicio. Estos componentes colaboran para proporcionar a cada proceso de usuario una interfaz distinta de la interfaz de llamadas al sistema Windows 2000. La tarea de las DLLs y de los subsistemas de entorno es implementar la funcionalidad de la interfaz publicada. Mientras que la interfaz Win32 es la interfaz oficial, las interfaces POSIX y OS/2 son, en la práctica, subsistemas inútiles por sus limitaciones de funcionalidad, cuya existencia obedece más a razones de imagen o imposición externa (caso de POSIX, por el gobierno americano) que a una intención de apertura por parte de Microsoft.

Dado que el enlace estático de los programas de usuario con las bibliotecas de la API Win32 conllevaría un tamaño enorme en los programas y un desperdicio de memoria, pues cada programa en ejecución tendría su copia de estas bibliotecas, todas las versiones de Windows manejan bibliotecas compartidas, llamadas bibliotecas de vínculos dinámicos (DLLs; Dinamic Link Libraries). En la figura se muestran algunas de las DLLs más importantes, con indicación del número de funciones de interfaz que publican y el modo en que se ejecutan.

Archivo	Modo	Funcs	Contenido
hal.dll	Kernel	95	Administración de hardware de bajo nivel, por ejemplo, E/S puertos.
ntoskrnl.exe	Kernel	1209	Sistema Operativo de Windows 2000 (kernel+ejecutivo).
win32k.sys	Kernel	-	Muchas llamadas al sistema, incluidas casi todas las de gráficos.
ntdll.dll	Usuario	1179	Despachador de modo de usuario a modo de kernel.
csrss.exe	Usuario	0	Proceso de subsistema de entorno Win32.
kernel32.dll	Usuario	823	Casi todas las llamadas al sistema centrales (no de gráficos).
gdi32.dll	Usuario	543	Llamadas de fuentes, texto, color, pincel pluma, mapa de bits, paleta, dibujo, etc.
user32.dll	Usuario	695	Llamadas de ventana, icono, menú, cursor, diálogo, Portapapeles, etc.
advapi32.dll	Usuario	557	Llamadas de seguridad, criptografía, Registro, administración.

## El Registro de Windows.

Windows necesita mantenerse al tanto de una gran cantidad de información acerca del hardware, el software y los usuarios. en Windows 3.x esta información se almacenaba en cientos de archivos .ini (de iniciación) dispersos por todo el disco. A partir de Windows 95, casi toda la información necesaria para arrancar y configurar el sistema y adaptarlo al usuario actual se reunió en una gran base de datos central llamada Registro (registry).

El registro consiste en una colección de directorios, llamados claves, cada uno de los cuales contiene subdirectorios (también claves) y entradas, llamadas valores.

Los valores o entradas contienen la información, cada valor tiene tres partes: un nombre, un tipo y los datos. El nombre no es más que una cadena en Unicode, el tipo identifica los datos, pudiendo utilizar alguno de estos 7 tipos de datos estándar:

REG_BINARY	almacena datos binarios arbitrarios, sin ningún tipo de reformato o análisis.
REG_DWORD	guarda un valor entero largo de 32 bits (o palabra doble). Se emplea normalmente cuando una entrada indica una cuenta o intervalo.
REG_SZ	almacena una cadena de caracteres ordinaria Unicode. Puede tener cualquier longitud.
REG_EXPAND_SZ	al igual que REG_SZ pero admite variables de sustitución predefinidas del sistema, identificadas como %variable%. El sistema realiza la sustitución cuando se solicita el valor almacenado.
REG_MULTI_SZ	es una colección, en un número arbitrario, de cadenas REG_SZ.

REG\_FULL\_RESOURCE\_DESCRIPTOR es un tipo no utilizado por los usuarios directamente, codifica información sobre los recursos de sistema requeridos por un dispositivo concreto.

REG\_NONE es solo un contenedor, se utiliza para permitir la existencia de un valor de registro que no contenga ninguna información.

En la raíz del registro, hay 5 claves predefinidas:

HKEY\_LOCAL\_MACHINE (HKLM) almacena todos los parámetros pertenecientes a la máquina local.

HKEY\_USERS (HKU) contiene una entrada por cada usuario que haya iniciado una sesión previamente en el ordenador. Cada entrada contiene los parámetros de configuración específicos de ese usuario.

HKEY\_CURRENT\_CONFIG (HKCC) guarda información relativa a la configuración actual del inicio del sistema. Esta clave predefinida es en realidad un apuntador o enlace a una subsección dentro de HKLM.

HKEY\_CURRENT\_USER (HKCU) es un apuntador o enlace al perfil de usuario almacenado en HKU, para el usuario que haya iniciado la sesión actual.

HKEY\_CLASSES\_ROOT (HKCR) enlaza las extensiones de archivo y los identificadores de clases OLE. Es un apuntador o enlace a HKLM\Software\Classes.

Desde estas claves principales, se desarrolla una estructura jerárquica de claves y valores similar a los directorios y archivos del sistema de ficheros.

### 3.2. PROCESOS Y THREADS.

El administrador de procesos es uno de los componentes del ejecutivo, dentro del sistema operativo. Cada proceso contiene al menos un subproceso o hebra (thread), el cual contiene al menos una fibra (subproceso ligero). Los procesos se pueden agrupar en trabajos para ciertos fines de administración de recursos, pues en Windows 2000 se han implementado, a nivel del sistema operativo las cuotas de procesador y la contabilidad de procesador, pudiendo por un lado limitar el uso excesivo del procesador, por parte de ciertos trabajos (conjunto de procesos) y por otra llevar cuenta de los recursos de proceso consumidos.

Los procesos se inician con una llamada al sistema con el nombre de un ejecutable. Cada proceso se inicia con un subproceso, los subprocesos son la base de la planificación de la CPU. El sistema operativo escoge el subproceso a ejecutar, no el proceso. El subproceso es un concepto de planificación, los recursos están a nivel de proceso. Cuando terminan todos los subprocesos de un proceso, éste termina. La conmutación de subprocesos es costosa pues requiere ingresar en modo kernel y salir luego de él.

El límite de 32 CPUs como máximo para el Windows 2000 en multiproceso, es fijo porque se usan varios mapas de bits para llevar el control de consumo interno de CPU de una sola palabra (la versión de 64-bit debería poder manejar hasta 64 procesadores).

Windows 2000 no cuenta con un subproceso planificador central. Cuando un subproceso no puede seguir ejecutándose, ingresa en modo kernel y ejecuta el planificador, él mismo, para ver cuál sub-

proceso cambiar. Las condiciones para que un subproceso que se está ejecutando active el código de planificación son:

1. El subproceso se bloquea por un semáforo, mutex, suceso, E/S, etc.
2. El subproceso envía una señal a un objeto.
3. El cuanto del subproceso se agota.

El planificador también se puede invocar cuando se cumplen otras dos condiciones:

1. Termina una operación de E/S.
2. Expira un plazo de espera.

Windows 2000 es apropiativo en su funcionamiento.

El algoritmo de planificación está basado en prioridades con alguna modificación dinámica de las mismas.

Cada proceso recibe una prioridad base para todos sus subprocesos. Los valores permitidos son: tiempo real, alta, mayor que la normal, normal, menor que la normal e inactiva. A su vez, cada subproceso puede variar su prioridad relativa al resto de subprocesos de un proceso en los siguientes valores: tiempo crítico, más alta, mayor que la normal, normal, menor que la normal, más baja e inactiva. En total un subproceso puede tener una prioridad de las 42 combinaciones posibles.

El sistema se basa en 32 prioridades del 0 al 31, se establece una correspondencia entre las 42 combinaciones anteriores y estas 32 prioridades, obteniéndose la prioridad base del proceso. Las prioridades 16 a 31 están reservadas al sistema operativo y a procesos colocados explícitamente por el administrador o usuario autorizado.

Las prioridades 1 a 15 son las reservadas para los procesos de usuario. la prioridad 0 es para el subproceso cero que se ejecuta en segundo plano y tiene como misión exclusiva rellenar de ceros las páginas que asignará el administrador de memoria. Existe un subproceso, denominado subproceso inactivo con prioridad -1 que se ejecuta cuando no se puede ejecutar ningún otro proceso en el sistema.

Las prioridades forman una tabla cuyas entradas apuntan a listas enlazadas de subprocesos de igual prioridad. El planificador funciona accediendo a la tabla por el proceso de prioridad 31 y viendo si tiene subprocesos listos para ejecutar. Si los hay toma el primero de la lista y lo ejecuta durante un cuanto de tiempo. Al finalizar coloca el proceso al final de la lista y toma el siguiente de la misma prioridad, de manera que, dentro de una misma prioridad sigue un algoritmo Round Robin. Mientras existan procesos preparados de una prioridad superior, el sistema les concederá todo el tiempo que precisen. Este comportamiento se repite para cada una de las entradas de la tabla de prioridades.

En ciertas condiciones un subproceso puede ver incrementada su prioridad base, pero nunca por encima de la prioridad 15 y nunca para subprocesos de prioridad mayor de 15. Si una operación de E/S libera un subproceso, este ve incrementada su prioridad base de modo que pueda ejecutarse pronto. También se produce aumento de prioridad si el subproceso estaba esperando por un semáforo, mutex u otro suceso. Estas elevaciones de prioridad van disminuyendo a medida que uno subproceso be-



neficiado va consumiendo por completo su cuanto de tiempo, hasta volver a situarse en su prioridad base. Existe un caso especial en el cual dos subprocesos trabajando juntos en un esquema productor-consumidor, donde el productor tiene una prioridad mayor que el consumidor, alcanzan un estado en el que el productor, se encuentra bloqueado por la inactividad del consumidor. Si aparece un subproceso cualquiera de prioridad intermedia, que no permite la ejecución del subproceso consumidor se produce una situación que Windows 2000 resuelve de una forma poco elegante: el sistema lleva cuenta del tiempo que un proceso listo lleva sin ejecutarse, si este tiempo excede de un cierto umbral, el subproceso se pasa a prioridad 15 durante dos cuantos de tiempo y después se devuelve, de manera abrupta, a su prioridad inicial. A este problema se le conoce como inversión de prioridad.

Windows 2000 permite ajustar el valor del cuanto por el administrador, entre dos posibles valores: priorizar las aplicaciones (interactivo) o los servicios (segundo plano). Priorizar las aplicaciones conlleva el empleo de cuantos más pequeños de tamaño variable; priorizar los servicios supone emplear cuantos más grandes de tamaño fijo.

### 3.3. GESTIÓN DE MEMORIA.

El administrador de memoria, componente del ejecutivo, implementa la arquitectura de memoria virtual paginada por demanda de Windows 2000.

En Windows 2000, cada proceso tiene su propio espacio de direcciones virtual de 32 bit, por lo que tiene 4 GB de espacio de direcciones. Los 2 GB inferiores están disponibles para el código y los datos de proceso; los 2 GB superiores se hacen corresponder con la memoria del kernel en forma protegida. El espacio de direcciones virtual se pagina por demanda con tamaño fijo de páginas (4 KB en el Pentium). No se maneja segmentación en ninguna de sus modalidades.

Todos los procesos de usuario comparten los 2 GB superiores, con excepción de las tablas de página, que son exclusivas de cada proceso.

Windows 2000 utiliza un algoritmo de paginación doble, por demanda y anticipada, leyendo un número variable de páginas según que el fallo se produzca en una página de código (lee más) o de datos.

El mecanismo de paginación se apoya mucho en el concepto de conjunto de trabajo. Cada proceso (no subproceso) tiene un conjunto de trabajo que consiste en las páginas con correspondencia que están en la memoria y a las que, por tanto, es posible acceder sin provocar un fallo de página. Las páginas de memoria real (mejor dicho los marcos de página) o están en un conjunto de trabajo o se encuentran en alguna de las cuatro listas de páginas libres: de páginas modificadas, de páginas en reserva, de páginas libres o de páginas en ceros. Existe una quinta lista de páginas de RAM defectuosas donde se mantienen las páginas con algún defecto para que no se empleen en ningún proceso. Diferentes subprocesos del sistema se encargan del mantenimiento de esta lista, buscando optimizar y minimizar la paginación a disco.

### 3.4. ENTRADA/SALIDA.

El administrador de E/S, componente del ejecutivo, proporciona un marco para administrar dispositivos de E/S y presta servicios genéricos de E/S. Este componente proporciona al resto del sistema E/S independiente del dispositivo. También es el lugar donde residen todos los controladores de dispositivos. Desde un punto de vista técnico, los sistemas de archivos son controladores de dispositivos

bajo el control del administrador de E/S. Existen dos sistemas de archivos distintos para los sistemas FAT y NTFS, independientes entre sí y cada uno controlando diferentes particiones del disco.

El administrador de E/S colabora estrechamente con el administrador de Plug-and-Play. La idea básica del Plug-and-Play es la de un Bus enumerable de tal modo que, el administrador de Plug-and-Play pueda enviar una solicitud a cada ranura pidiendo al dispositivo que está ahí que se identifique. Una vez que ha descubierto qué dispositivos están conectados, dicho administrador asigna recursos de hw, localiza los controladores apropiados y los carga en la memoria.

Una característica interesante de Windows 2000 es su manejo de volúmenes dinámicos que pueden abarcar múltiples particiones e incluso múltiples discos y reconfigurarse sobre la marcha, sin necesidad de reiniciar el sistema. Otro aspecto importante es la posibilidad de manejo de E/S asíncrona. Un subproceso puede iniciar una operación de E/S y luego seguir ejecutándose en paralelo con la E/S.

El nuevo modelo de controladores de Windows 2000 (WDM Windows Driver Model) permite que los sistemas operativos Windows 2000 y Windows 98 utilicen los mismos controladores, lo que supone una ventaja para los fabricantes, los administradores e incluso los usuarios.

### 3.5. EL SISTEMA DE ARCHIVOS.

Windows 2000 reconoce varios sistemas de archivos, siendo los más importantes FAT-16, FAT-32 y NTFS (NT File System). FAT-16 es el antiguo sistema de archivos de MS-DOS; usa direcciones de disco de 16 bits, lo que lo limita a particiones de disco de un máximo de 2 GB. FAT-32 emplea direcciones de disco de 32 bits y maneja particiones de disco de hasta 2 TB. NTFS es un nuevo sistema de archivos, creado específicamente para Windows NT y que se ha trasladado a Windows 2000. Utiliza direcciones de disco de 64 bits y puede (en teoría) manejar particiones de hasta 264 bytes. Windows 2000 también reconoce sistemas de solo lectura para CD-ROM y DVD. Windows 2000 ha dejado de soportar el sistema HPFS que mantenía por compatibilidad con el sistema operativo OS/2.

Con la nueva versión del sistema de archivos NTFS 5, se han incorporado nuevas funcionalidades como la incorporación de cuotas de disco, el sistema de cifrado de archivos (EFS Encrypting File System), si bien esta funcionalidad la realiza un controlador, con este nombre, que está situado entre el NTFS y el proceso de usuario, las capacidad nativa de desfragmentación de volúmenes y el Servicio de seguimiento de enlaces distribuidos, que asegura a los usuarios finales la capacidad de encontrar archivos por la red que han sido movidos de sus ubicaciones originales por los administradores.

#### NTFS.

En NTFS, los nombres de archivo individuales están limitados a 255 caracteres; las rutas completas se limitan a 32.767 caracteres. Los nombres de archivo están en Unicode. NTFS distingue entre mayúsculas y minúsculas, pero la API Win32 no reconoce bien esta diferencia, y nunca en los directorios.

Un archivo NTFS no es solo una sucesión lineal de bytes, como los archivos FAT-32 y UNIX. Más bien, un archivo consiste en múltiples atributos, cada uno de los cuales se representa con un flujo de bytes. Casi todos los archivos tienen unos cuantos flujos cortos, como el nombre del archivo y su identificador de objeto de 64 bits, y un flujo largo (sin nombre) que contiene los datos. Sin embargo, un archivo puede tener dos o más flujos de datos (largos). Cada flujo tiene un nombre que consta del nombre del archivo, un signo de dos puntos y el nombre del flujo, por ejemplo archivo:flujo. Cada flujo tiene su

propio tamaño y se puede bloquear con independencia de todos los demás flujos. Esta idea se tomó de la Apple Macintosh, donde los archivos tienen dos flujos: la rama de datos y la rama de recursos.

La principal estructura de datos de cada volumen es la tabla maestra de archivos MFT (Master File Table), que es una sucesión lineal de registros de tamaño fijo (1 KB). Cada registro de MFT describe un archivo o un directorio, contiene los atributos del archivo, como su nombre y marcas de tiempo, y la lista de direcciones de disco donde están sus bloques. Si un archivo es demasiado grande puede ser necesario emplear más de un registro MFT para contener la lista de todos los bloques. En este caso al primer registro se le denomina registro base, y apunta a los demás registros MFT.

Como la MFT es un archivo, puede ser colocada en cualquier lugar del disco, por lo que no está limitada a la primera pista del disco. Cada registro MFT es una secuencia de pares (encabezado de atributo, valor).

Windows 2000 incluye como novedad el Sistema de Archivos Distribuido (DFS), que permite a un administrador crear un árbol de directorios que consista en particiones de distintos sistemas de la red. El DFS soporta tolerancia a fallos y balanceo de carga, así como replicación para mantener sincronizados los datos.

### 3.6. SEGURIDAD.

El administrador de seguridad, componente del ejecutivo, hace que se respete el complejo mecanismo de seguridad de Windows 2000, que satisface los requisitos C-2 del Libro Naranja del Departamento de Defensa de Estados Unidos. El Libro Naranja especifica un gran número de reglas que debe cumplir un sistema, desde validar los inicios de sesión hasta administrar el control de acceso y llenar con ceros las páginas virtuales antes de reutilizarlas.

Cada usuario y grupo de Windows 2000 se identifica con un SID (Security ID) único a nivel mundial. Cada proceso lleva asociado una ficha de acceso que especifica su SID y otras propiedades.

Cada objeto tiene asociado un descriptor de seguridad que indica quién puede realizar qué operaciones con él. Un descriptor de seguridad está formado por un encabezado, seguido de una DACL (discretionary Access Control List) con uno o mas elementos de control de acceso (ACE). Los más importantes son Allow y Deny. Además del DACL, el descriptor tiene una SACL (System Access Control List) que no especifica quién puede usar el objeto sino qué operaciones con el objeto se asientan en el registro de sucesos de seguridad del sistema (función de auditoría).

En un sistema autónomo la validación corre por cuenta del proceso winlogon y la configuración de seguridad almacenada en la propia máquina en las claves del registro: SECURITY y SAM. Donde la primera establece las políticas globales de seguridad y la segunda la seguridad específica de cada usuario.

En un sistema en red, la autenticación de los usuarios está centralizada en ciertos servidores denominados controladores del dominio. Los equipos se organizan dentro de Dominios, pudiendo éste estar gestionado mediante el empleo del Active Directory.

Windows 2000 es el primer sistema de Windows que dispone de administración centralizada de certificados. En las versiones anteriores se confiaba que cada aplicación mantenía su propia lista de claves o CA confiables.

El protocolo KERBEROS (RFC 1510), que es un estándar de Internet para autenticación, es el método nativo que emplean los sistemas Windows 2000. Cualquier servidor del Directorio Activo, automáticamente, tiene el servicio del Centro de distribución de claves de Kerberos (KDC- Kerberos Key Distribution Center).

## 4. LOS INTÉRPRETES DE COMANDOS.

### 4.1. EL SHELL.

Aunque muchos sistemas UNIX tienen una interfaz gráfica de usuario, del tipo que se popularizó gracias a los ordenadores Macintosh y posteriormente con Windows, los verdaderos programadores siguen prefiriendo una interfaz de línea de comandos llamada shell. Su uso es más rápido, más potente y puede extenderse con facilidad.

El shell no es parte del núcleo del sistema operativo, funciona en modo usuario lo que permite ser reemplazado con facilidad, existiendo hoy en día una gran variedad de shells.

Cuando un shell se pone en marcha, se inicializa y luego muestra en la pantalla un indicador de comandos (prompt), que suele ser un signo de porcentaje o dólar, y espera hasta que el usuario teclee una línea de comando, cuyo fin se reconoce con la pulsación de la tecla ENTER. Cuando el usuario termina, el shell extrae la primera palabra de la línea, supone que es el nombre de un programa a ejecutar, lo busca y, si lo encuentra, lo ejecuta. En ese momento el shell se suspende a sí mismo hasta que el programa termina, entonces trata de leer la siguiente línea de comandos.

Los comandos pueden tener argumentos, que se pasan al programa invocado en forma de cadenas de caracteres. Por ejemplo, la línea de comandos:

```
cp origen destino
```

invoca el programa cp (copiar) con dos argumentos origen y destino. Este programa interpreta el primero como el nombre de un archivo existente, crea una copia de ese archivo y le asigna el nombre destino.

Una vez invocada, la shell presenta unas características que la hacen muy parecida a un idioma de programación interpretado:

- Variables.
- Estructuras de control if, while...
- Subprogramas.
- Paso de parámetros.
- Manejo de interrupciones.

Estas características le dotan de capacidad para diseñar herramientas propias, mediante el empleo de archivos de comandos, denominados shell-scripts.

Todo comando o shell-script en ejecución tiene asociado tres ficheros estándar: stdin, stdout y stderr. Es posible redireccionar el origen o el destino de estos ficheros (inicialmente asociados a la entrada estándar o teclado y a la salida estándar o terminal) (>,<>>,<<); secuenciar comandos (;); encolar comandos de modo que la salida de uno sea la entrada del siguiente (|) (pipe o tubería); ejecutar multiproceso (&).

Cualquier mejora en el shell puede repercutir en un uso más efectivo del sistema e incluso permitir realizar cosas imposibles hasta entonces.

La primera mejora que la mayoría de los nuevos shells aportan es el incremento en la velocidad. Se necesita teclear menos para conseguir el mismo resultado, derivado de las capacidades de auto-completar, del incremento de la información suministrada (p. ej. mostrando el directorio actual como parte del prompt), cubriendo alguna de las carencias del propio UNIX como el no retroceder a través de los enlaces simbólicos entre directorios.

## 4.2. HISTORIA DE LOS SHELL DE UNIX-LINUX.

Inicialmente, el primer shell fué el Bourne Shell /bin/sh, escrito por S.R. Bourne. Tenía incorporado, y sigue teniendo, un lenguaje sintáctico muy potente y robusto, con todas las características que normalmente se necesitan para generar programas estructurados. En particular, dispone de robustas capacidades de control de la entrada y salida y del tratamiento de concordancia de expresiones. Sin embargo, al margen de lo robusto del lenguaje, tiene un gran inconveniente: prácticamente no da ninguna concesión al usuario interactivo (la única concesión real ha sido el uso de funciones de shell y es algo que se incorporó más tarde).

Más tarde, en la universidad de Berkeley en California se desarrolla el C Shell /bin/csh. Este shell incluyó varios conceptos nuevos (la mayoría sobre control de trabajos y alias) y procuró realizar un shell que fuera mucho mejor para el uso interactivo. Al tiempo que buscaba un mejor comportamiento interactivo, se decantaron por un lenguaje de comandos diferente. La teoría era semejarse a C el lenguaje en que estaba escrito el propio UNIX, sin embargo el resultado fue un sistema tan lleno de errores que era incapaz de generar shell-scripts robustos y fiables, por lo que todo el mundo mantuvo el Bourne Shell para esa tarea, pero a la vez era mucho mejor para el uso interactivo por lo que se llegó a una situación en la que un usuario empleaba un shell diferente para uso interactivo y para no interactivo.

Después de que el csh se fuera abandonando, varias personas decidieron que se debían corregir los errores, y mientras se corregían, incorporar características nuevas. El resultado fue la edición de la línea de comando, el autocompletado estilo-TENEX y varias características más, pasándose a denominar tcsh. Sin embargo los fabricantes siguieron suministrando el C Shell estándar añadiéndole características no estándar, conforme iban saliendo.

Sobre este tiempo, David Korn, de AT&T, tuvo la idea de ordenar todo el barullo, apareciendo el Korn Shell /bin/ksh, este recortó el lenguaje de los C Shells volviendo al lenguaje del Bourne Shell, aunque añadió la mayoría de características que hacían del C shell bueno para el trabajo interactivo. El Korn Shell pasó a ser parte del sistema operativo System V, pero tenía un gran problema: a diferencia del resto de shells, no era libre y se debía pagar a AT&T por su uso.

Sobre estas fechas, comienzan los primeros intentos de estandarizar UNIX mediante los estándar POSIX. En un principio, la especificación POSIX era más o menos el Bourne Shell, más tarde el estándar se actualizó, asemejándose más al ksh.

Por este tiempo, el proyecto GNU estaba en marcha, decidiéndose la necesidad de un shell libre compatible con POSIX, dando lugar al nacimiento de bash (Bourne Again Shell). A semejanza del Korn Shell, bash se basó en el lenguaje del Bourne Shell y, también como el Korn Shell, tomó características del C Shell y otros sistemas operativos, sin embargo, la gran diferencia es que bash es libre. Bash fue pronto adoptado por la comunidad LINUX (pudiéndose configurar para que funcione como el Bourne Shell), siendo en la actualidad el más popular entre la nueva generación de shells libres.

Tom Duff, que se encontraba en la tarea de portar el Bourne Shell a Plan 9 (Sistema Operativo distribuido de los laboratorios Bell), se «sublevó» y en lugar de portar el shell, creó un nuevo shell el rc. Después de publicar un artículo sobre él, Byron Rakitzis lo desarrolló para UNIX. Con la ventaja de partir de cero, rc resultó un shell pequeño, más simple, estable y, en opinión de muchos, mucho más limpio.

La búsqueda del shell perfecto todavía continúa, el último en entrar ha sido zsh. Zsh fué escrito por Paul Falstad, estudiante de la universidad de Princeton, está basado principalmente en el Bourne Shell aunque con algunas diferencias pequeñas pero importantes, Incluyendo una gran cantidad de características adicionales. Por su parte rc ha sido mejorado dando lugar a es, este shell incorpora la posibilidad de redefinir funciones de bajo nivel por parte del usuario.

#### 4.3. COMPARATIVA ENTRE SHELLS.

Característica	sh	csch	ksh	bash	tcsch	zsh	rc	es
Control de trabajos.....	N	S	S	S	S	S	N	N
Alias .....	N	S	S	S	S	S	N	N
Funciones Shell.....	S(1)	N	S	S	N	S	S	S
Redirección de Entrada/Salida «sensible» .....	S	N	S	S	N	S	S	S
Pila de directorios .....	N	S	S	S	S	S	F	F
Histórico de comandos .....	N	S	S	S	S	S	L	L
Edición de la línea de comando .....	N	N	S	S	S	S	L	L
Edición de la línea de comando con Vi.....	N	N	S	S	S(3)	S	L	L
Edición de la línea de comando con Emacs.....	N	N	S	S	S	S	L	L
Búsqueda de nombre de usuarios .....	N	S	S	S	S	S	L	L
Autocompletado de nombres de archivo .....	N	S(1)	S	S	S	S	L	L
Autocompletado de nombres de usuario.....	N	S(2)	S	S	S	S	L	L
Autocompletado de nombres de equipos .....	N	S(2)	S	S	S	S	L	L
Histórico de auto completados.....	N	N	N	S	S	S	L	L
Autocompletado totalmente programable.....	N	N	N	N	S	S	N	N
Evaluación aritmética incorporada .....	N	S	S	S	S	S	N	N
Puede seguir enlaces simbólicos transparentemente.	N	N	S	S	S	S	N	N

.../...

.../...

Ejecución periódica de comandos .....	N	N	N	N	S	S	N	N
Prompts a medida (fácilmente) .....	N	N	S	S	S	S	S	S
Corrector ortográfico .....	N	N	N	N	S	S	N	N
Sintaxis en que se basa .....	sh	cs	sh	sh	cs	sh	rc	rc
Disponible libremente .....	N	N	N(4)	S	S	S	S	S
Comprueba el buzón de correo .....	N	S	S	S	S	S	F	F
Comprueba la corrección de los terminales .....	N	N	N	N	S	S	N	N
Puede manejar listas largas de argumentos .....	S	N	S	S	S	S	S	S
Dispone de archivo de inicialización no interactivo.	N	S	S(5)	S(5)	S	S	N	N
Puede evitar archivos de inicialización de usuario ..	N	S	N	S	N	S	S	S
Puede especificar un archivo de inicialización .....	N	N	S	S	N	N	N	N
Redefinición de funciones de bajo nivel .....	N	N	N	N	N	N	N	S
Dispone de funciones anónimas .....	N	N	N	N	N	N	S	S
Lista de Variables .....	N	S	S	N	S	S	S	S
Completa gestión y manejo de señales .....	S	N	S	S	N	S	S	S
Variables locales .....	N	N	S	S	N	S	S	S
Excepciones .....	N	N	N	N	N	N	N	S
<b>Leyenda</b>								
S    Característica implementada en el shell								
N    Característica no disponible en el shell								
F    Característica disponible a través del mecanismo de funciones del shell								
<b>Notas</b>								
1    Esta característica no se encontraba en el original pero después se ha convertido prácticamente en estándar.								
2    Esta característica es prácticamente nueva y a menudo no se encuentra en muchas versiones del shell, pero se va haciendo paso gradualmente en la distribución estándar.								
3    La emulación del vi en este shell, es vista por muchos como incompleta.								
4    Una versión denominada pdksh está disponible libremente, pero no tiene la funcionalidad completa de la versión de AT&T.								
5    Solamente especificando un archivo mediante la variable de entorno ENV.								

La tabla anterior muestra muchas de las características que pueden servir para elegir el shell más apropiado. La lista no es exhaustiva ni definitiva, se ha considerado como característica incluida en un

shell, cuando está disponible en la versión suministrada con el sistema operativo o por compilación directa de la distribución estándar. En particular, las características del C Shell son las disponibles en la versión SunOS 4.\*

#### 4.4. EL INTÉRPRETE DE COMANDOS DE WINDOWS.

Mientras que UNIX y LINUX nacieron como sistemas operativos cuya interfaz hombre-máquina era ofrecida por el shell a través de su línea de comandos, y posteriormente se incorporó una GUI para mejorar esa interfaz, Windows nace como evolución del MS-DOS siendo un sistema operativo de entorno gráfico completamente integrado con el resto de las funcionalidades del sistema.

Hasta Windows 98, el equivalente al intérprete de comandos de UNIX, era en realidad el sistema operativo MS-DOS, con su intérprete de línea de comandos command.com. En las versiones sucesivas de Windows, se ha incorporado un intérprete nuevo cmd.exe que ya no guarda ninguna relación con el antiguo MS-DOS, si bien se ha mantenido una compatibilidad relativa con él.

A través del programa cmd.exe se accede a una ventana de línea de comandos, similar al antiguo MS-DOS, que tiene implementados un reducido número de comandos del antiguo sistema operativo. Aunque se permite la creación y ejecución de archivos de comandos (con la extensión .bat), ya desde Windows98 Microsoft incorporó una herramienta nueva creación y ejecución de archivos de comandos o scripts denominada WSH Windows Script Host.

WSH, crea un entorno para la ejecución de scripts, independiente del lenguaje empleado en los mismos (VBScript, JScript, etc.), proporciona los servicios necesarios para el archivo de comandos, gestionando la seguridad y encargándose de invocar el intérprete adecuado al lenguaje de script empleado.

WSH representa la respuesta de Microsoft a las necesidades de contar con un sistema de automatización de tareas, similar al ofrecido por los Shell de UNIX, LINUX.

### 5. ESTRUCTURA DE ARCHIVOS.

#### 5.1. LOS ARCHIVOS DE UNIX, LINUX.

En UNIX, un fichero es una secuencia de 0 o más bytes. No existiendo distinción entre ficheros ASCII, binarios, etc.

Se admiten nombres de hasta 255 caracteres. No siguiendo el esquema habitual de nombre.extensión

El sistema de archivos de UNIX, desde el punto de vista del usuario, tiene una organización jerárquica o de árbol invertido que parte de una raíz conocida como «/» (diagonal). Es una diagonal al revés que la usada en DOS. Hay dos formas de especificar nombres de archivo, tanto en el shell como al abrir un archivo desde dentro de un programa. La primera es usar una ruta absoluta, que supone indicar cómo llegar al archivo partiendo del directorio raíz. Un ejemplo de ruta absoluta es /home/usua-



rio/personal/diario. Esto le dice al sistema que el fichero diario se encuentra desde el directorio raíz pasando por el directorio home que tiene un directorio usuario que a su vez tiene un directorio personal donde se encuentra el archivo diario buscado.

Como los nombres de ruta absoluta suelen ser largos e inconvenientes, UNIX permite designar el directorio en el que se está trabajando en un momento dado, como directorio de trabajo. Los archivos entonces pueden designarse en relación al directorio de trabajo, lo que se conoce como ruta relativa. Por ejemplo, si un usuario se encuentra en el directorio /home/usuario/personal entonces el comando shell:

```
cp diario DiarioDeAyer
```

tiene el mismo efecto que el comando:

```
cp /home/usuario/personal/diario /home/usuario/personal/DiarioDeAyer
```

Por otro lado, si un usuario necesita referirse a un archivo que se encuentra ubicado en otro lugar del árbol de archivos, alejado del directorio de trabajo actual, será necesario usar un nombre de ruta absoluto o cambiar el directorio de trabajo, lo que no siempre es aceptable. Si el nombre es muy largo, podría resultar molesto y engorroso. UNIX permite que los usuarios creen una entrada nueva de directorio que apunte a un archivo o directorio existente en otra parte de la estructura. Tal entrada se denomina enlace. Existen dos tipos de enlaces: el enlace tal cual, que es una entrada de directorio que enlaza directamente con el fichero de datos y el enlace simbólico que es una entrada que apunta a una ruta absoluta a donde se dirigirá la búsqueda de un archivo determinado. Un enlace solamente puede referirse a archivos dentro del mismo sistema físico de archivos; un enlace simbólico puede referirse a una ruta cualquiera en todo el árbol de directorios, incluso a una ruta inexistente.

Si dos usuarios distintos tienen sendos enlaces a un archivo, la operación de borrado del mencionado archivo por parte de uno de ellos, solamente supondrá la eliminación de la entrada de directorio del usuario que borra. El otro usuario mantendrá el archivo intacto. Cuando el segundo usuario elimina a su vez el archivo, el sistema procede a eliminar la entrada de directorio correspondiente y, al comprobar que ya no quedan más enlaces al archivo en cuestión, procede a su borrado definitivo.

En UNIX, la estructura jerárquica de archivos se organiza en base a un único nodo raíz. Mientras que los sistemas operativos de la familia MS-DOS o WINDOWS, se organizan de forma jerárquica dentro de cada partición de disco físico creada, dando lugar a tantos nodos raíz como particiones se hallan conectado, nodos conocidos por una letra desde la A hasta la Z, las diferentes particiones de los discos en UNIX, se insertan a lo largo de la estructura jerárquica del directorio, en puntos de enlace que se presentan como un subdirectorio cualquiera. De esta forma, el usuario no conoce ni debe preocuparse en saber dónde se almacenará físicamente la información, sino en qué parte del sistema de archivos debe buscarla.

Dentro de la estructura de directorios de UNIX existen una serie de directorios comunes a todas las instalaciones que es preciso conocer:

## 5.2. LA JERARQUÍA DE DIRECTORIOS UNIX, LINUX.

/	directorio raíz, inicio del sistema de archivos
/bin	archivos ejecutables, comandos de usuario
/boot	archivos de arranque
/cdrom	punto de montaje para la unidad de CD-ROM
/dev	archivos especiales de dispositivos

### [subdirectorios propios de System V]

./dsk	dispositivos de disco
./fd	dispositivos descriptores de archivo
./kd	dispositivos de teclado y despliegue
./kmem	memoria
./null	dispositivo para descarte de salidas
./osm	mensajes de error del núcleo
./pts	pseudo ttys; igual que /dev/pts*
./rdsk	dispositivos crudos de disco
./term	terminales; igual que /dev/tty*
./xt	pseudo ttys; para capas DMD
/dosc	punto de montaje para la partición DOS
/etc	configuración de paquetes, configuración de sistema
./init.d	scripts de arranque y detención de programas
./rc?.d	enlaces a scripts, con K o S (Kill o Start), y número de secuencia para controlar el arranque
./skel	archivos de inicialización para nuevos usuarios
/export	directorios de usuarios en sistemas grandes
/floppy	para montar una unidad de disquete
/home	objetos relacionados con los usuarios
/lib	bibliotecas de desarrollo y material de apoyo
/lost+found	archivos perdidos
/mnt	punto de montaje de dispositivos externos
/proc	archivos de control de procesos
/root	directorio propio para el supervisor (root)

/sbin	archivos ejecutables de administración
/tmp	archivos temporales
/usr	ejecutables, documentación, referencia
./X11R6	sistema X-Windows
./bin	más ejecutables
./doc	documentos de paquetes de software
./include	encabezados .h de bibliotecas en C
./info	archivos de info, información de UNIX (GNU)
./lib	más bibliotecas en C
./local	ejecutables instalados por el administrador
./man	subdirectorios de páginas del manual
./sbin	más archivos ejecutables de administración
./share	compartidos
./src	(source) código fuente del kernel
/var	archivos de log, auxiliares, archivos que crecen
./backup	respaldo de algunos archivos del sistema
./catman	páginas man ya formateadas
./lib	información propia de programas
./lock	control de bloqueos
./log	archivos de registro de mensajes (log) del sistema
./spool	colas de impresión, intermedios de correo y otros /run información de procesos (PIDs)

Además de estos directorios, el administrador de cada sistema, creará la estructura adecuada a la Organización, a la vez que los diferentes productos y herramientas que se instalen crearán y mantendrán su propia jerarquía de directorios y archivos, que será preciso conocer por parte de los usuarios que deban interaccionar con ellas.

### 5.3. LOS ARCHIVOS DE WINDOWS.

A diferencia de UNIX o LINUX, Windows no mantenía una estructura jerárquica única para su sistema de archivos. Si bien el modelo jerárquico de directorios se mantiene, éste va asociado a la unidad lógica de almacenamiento.

Windows permite la división y unión (mediante RAID) de las unidades físicas de almacenamiento en unidades lógicas o particiones más pequeñas o mayores que la propia unidad física. Cada unidad

lógica se reconoce con una letra del alfabeto A-Z, estando reservadas las dos primera (A y B) para las unidades de disco flexible.

Para cada unidad lógica de almacenamiento, Windows crea una estructura jerárquica de directorios independiente. Para acceder a un cierto archivo es necesario conocer la ruta completa, que incluye la letra de la unidad en la que se encuentra almacenado.

El formato que sigue una ruta completa de archivo es:

<Letra de Unidad>:\<directorio>\...\<directorio>\<nombre de archivo>.<extension>

Así, un nombre como C:\Personal>Contactos\Agenda.txt representa al archivo Agenda.txt, ubicado en el directorio Contactos, que depende del directorio Personal dentro de la unidad lógica C.

Windows se instala por defecto en la primera unidad lógica de disco duro, o unidad C. dentro de la misma, crea una estructura jerárquica de directorios donde se ubica durante el proceso de la instalación, siendo los más representativos los siguientes:

C:\Windows

Este es el directorio principal del Sistema Operativo. Desde él se desarrolla toda la estructura jerárquica de directorios necesarios para el correcto funcionamiento del sistema.

C:\Archivos de Programa

Bajo este directorio, se instalan los programas y aplicaciones del usuario, que no son una parte fundamental del propio sistema operativo.

C:\Documents and Settings

A partir de Windows 2000, aparecen los perfiles de usuario que permiten personalizar la apariencia del sistema en función del usuario que inicie la sesión. Bajo este directorio se almacena toda la información particularizada de un usuario del sistema. Para cada usuario diferente del sistema (identificado por un nombre y una clave de acceso, al iniciar una sesión en la máquina) se genera un directorio con su nombre del que se cuelga toda la personalización del sistema así como la información privada. Así, en estos directorios se almacena la estructura de menús particular del usuario, la parte del registro de windows correspondiente a la rama HKEY\_CURRENT\_USER, el directorio temporal, los archivos temporales de internet, las carpetas personales de correo, el directorio Mis Documentos, el Escritorio que el usuario se encuentra al acceder al sistema, etc.

## 6. INTERFACES GRÁFICOS.

### 6.1. LOS ORÍGENES DEL GUI DE UNIX, LINUX.

El software de la interfaz gráfica de usuarios o GUI (Graphical User Interface), puede implementarse en el sistema operativo mismo, como se hace en Windows, o en código de nivel de usuario, como ocurre en los sistemas UNIX.

A diferencia de los sistemas MS-DOS y Windows que son sistemas monousuarios, pensados para funcionar en un ordenador personal, los sistemas UNIX son sistemas multiusuario, pensados para ser ejecutados en un sistema central, dando servicio a una variedad de usuarios, que acceden a él a través de terminales. En este entorno la ejecución a nivel de usuario del software GUI, permite que cada usuario disponga de su propia copia, evitando que un mal funcionamiento ponga en peligro la estabilidad de todo el sistema. Si un usuario tiene problemas con el GUI es suficiente con cancelar el programa y volver a ejecutarlo.

A mediados de 1980 se creó una fundación para entornos de usuario gráficos independientes del sistema operativo que se llamó X-Window. Las especificaciones X-Window definen el método por el cual se pueden comunicar las aplicaciones con el hardware gráfico. También establecen un conjunto de funciones de programación bien definidas que podrán ser llamadas para realizar la manipulación básica de las ventanas. X Window (conocido también como X), desarrollado en el MIT (Massachusetts institute of technology) como parte del proyecto Athena, ha constituido la base sobre la que se han desarrollado otros sistemas GUI posteriores.

El sistema X Window se basa en la disponibilidad de terminales inteligentes con una CPU tan potente como la del ordenador principal, memoria suficiente, un teclado y un ratón. Un terminal de este tipo se le conoce como Terminal X, y se encuentra ejecutando el X Window System. Este tipo de terminales interactúan con programas que se ejecutan en un ordenador remoto.

El programa, dentro del Terminal X, que obtiene entradas del teclado o del ratón y acepta comandos de un ordenador remoto, se denomina Servidor X. Este programa debe mantenerse al tanto de la ventana seleccionada en un momento dado (donde se encuentra el puntero del ratón), para saber a qué cliente debe enviar las entradas que lleguen del teclado.

El Servidor X se comunica a través de la red con Clientes X que se ejecutan en algún host (ordenador central) remoto. El servidor envía a los clientes entradas del teclado y el ratón y acepta comandos de despliegue de ellos.

Aunque parezca extraño que sea el servidor el que se coloque en la terminal, hay que tener en cuenta que la tarea del servidor es exhibir bits por lo que tiene sentido estar cerca del usuario. Desde el punto de vista del programa, se trata de un cliente que le dice al servidor que haga cosas, como exhibir texto y figuras geométricas. El servidor (en la terminal) simplemente obedece, al igual que todos los servidores. En la figura se muestra la disposición del cliente y el servidor.

Es posible ejecutar X Window System en UNIX o en otro sistema operativo. Muchos sistemas UNIX ejecutan X como su sistema de ventanas estándar, tanto en máquinas autónomas, como para acceder a ordenadores remotos por internet. Lo que X Window system define en realidad es el protocolo entre el Cliente X y el Servidor X, no importa si el cliente y el servidor están en la misma máquina, dentro de una red local o a miles de kilómetros conectados por internet. El protocolo y el funcionamiento del sistema es idéntico en todos los casos.

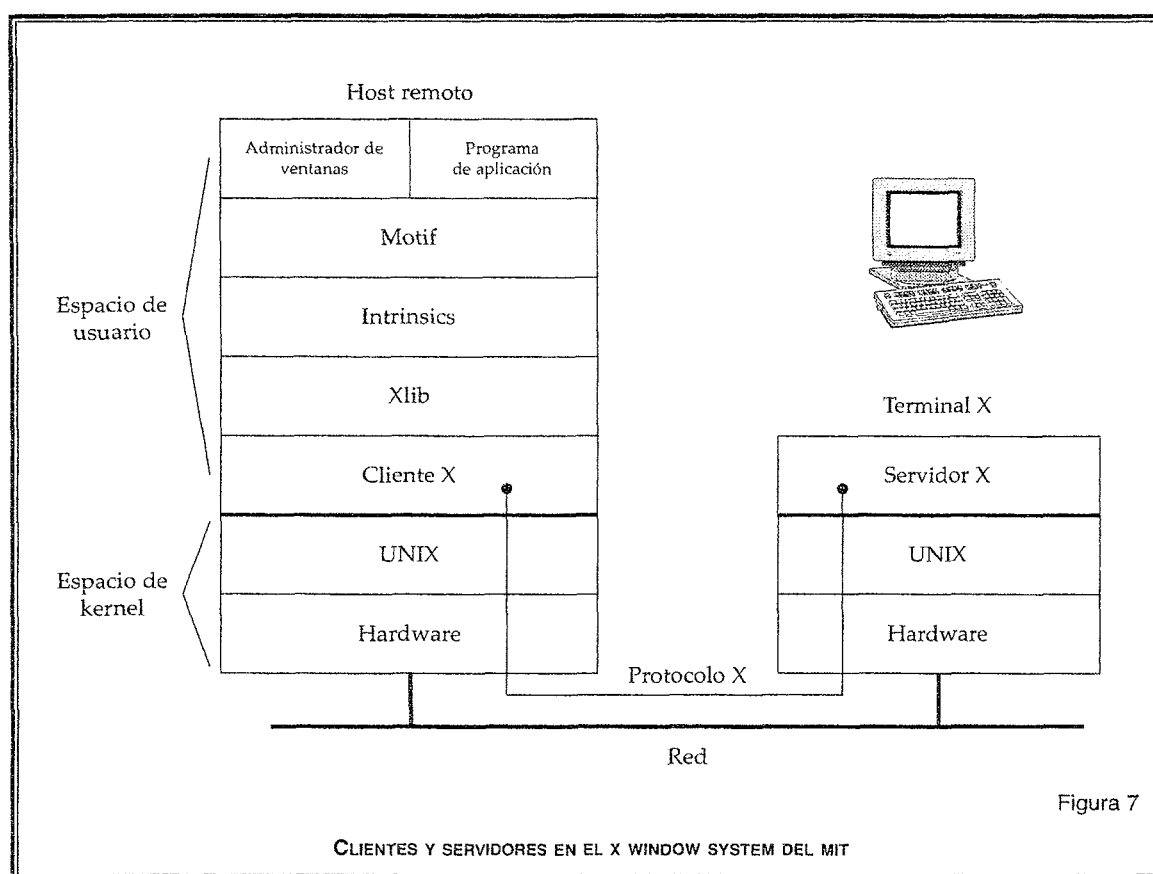
X no es más que un sistema de ventanas; no es una GUI completa. Para tener una GUI completa, sobre él se ejecutan otras capas de software. Una capa es Xlib, que es un conjunto de procedimientos de biblioteca para acceder a la funcionalidad X. Estos procedimientos constituyen la base de X Window System, pero son demasiado primitivos para que la mayoría de los programas de usuario accedan a ellos en forma directa.

A fin de facilitar la programación con X, se proporciona como parte de X un conjunto de herramientas llamado Intrinsics. Esta capa administra botones, barras de desplazamiento y otros elementos de una GUI, llamados widgets. Para tener una verdadera GUI, con aspecto y funcionamiento unifor-

mes, se necesita una capa más. La más popular se denomina Motif. La mayoría de las aplicaciones emite llamadas a Motif, no a Xlib.

También vale la pena señalar que la administración de ventanas no forma parte de X mismo. La decisión de no incluirla fue plenamente intencional. En vez de eso, uno proceso cliente X aparte, llamado administrador de ventanas, controla la creación, eliminación y movimiento de ventanas en la pantalla. Éste administra las ventanas enviando comandos al servidor X para indicarle qué debe hacer. El administrador de ventanas suele ejecutarse en la misma máquina que el cliente, pero en teoría puede hacerlo en cualquier lado.

Este diseño modular, que consta de varias capas y múltiples programas, hace que X sea muy portátil y flexible. El sistema se ha trasladado a casi todas las versiones de UNIX, incluidas Solaris, BSD, AIX, Linux, etc., de modo que los creadores de aplicaciones pueden disponer de una interfaz de usuario estándar para múltiples plataformas. X también se ha trasladado a otros sistemas operativos. En contraste, en Windows los sistemas de ventanas y de GUI están mezclados dentro de la GDI y situados en el kernel, lo que dificulta su mantenimiento. La característica más importante del entorno gráfico es su completa independencia del núcleo del sistema operativo, lo que permite seleccionar diferentes interfaces para trabajar.



## 6.2. OTROS GUI DE UNIX, LINUX.

La dificultad de programar en el entorno X Window propició la aparición, a finales de los 90, de dos grupos independientes que aportaron soluciones a estos problemas: GNOME y KDE. KDE ofrece

un gestor de ventanas nuevo y las bibliotecas necesarias para escribir aplicaciones de un modo mucho más fácil. GNOME ofrece un marco general para el resto de gestores de ventana y para las aplicaciones que trabajan con ellos. Cada uno tiene una idea diferente de cómo deben funcionar las cosas, pero debido a que trabajan por encima de X-Window, no son enteramente incompatibles.

KDE (K Desktop Environment) o entorno de escritorio K, es un entorno de escritorio ligeramente distinto de los gestores típicos de ventanas. En lugar de describir cómo se ve la interfaz, KDE proporciona un conjunto de bibliotecas que, si se usan, permiten a una aplicación mejorar algunas características especiales que ofrece el gestor de ventanas (soporte de pinchar y arrastra, soporte de impresión estandarizado, etc.). KDE ofrece al programador una biblioteca mucho más sencilla de usar que el trabajo directo con la interfaz X-Windows. Una de las herramientas más potentes es el escritorio virtual, lo que permite tener varias pantallas efectivas al mismo tiempo.

GNOME (GNU Network Object Model Environment) al igual que KDE ofrece un entorno de escritorio completo y un marco de aplicaciones para desarrollo tan bueno como de fácil uso. A diferencia de KDE, GNOME no es un gestor de ventanas, proporciona bibliotecas de desarrollo y características de gestión de sesión que nosotros, como usuarios, no vemos. En lo más alto de su estructura está un gestor de ventanas que muestra la apariencia general del escritorio. El gestor de ventana por defecto es Enlightenment, pero hay disponibles varias opciones más. Desde el punto de vista del desarrollador, GNOME define sus interfaces con el resto del mundo mediante tecnología CORBA. Cualquier sistema de desarrollo que pueda comunicarse usando CORBA puede utilizarse para desarrollar aplicaciones compiladas en GNOME.

En KDE, el escritorio está compuesto por seis partes principales: el Window Manager (kwin); el Desktop Manager (kdesktop); el Panel (kicker); el Control Center (kcontrol); el Help Center (khelp-center) y el Browser/File Manager/Universal Viewer (konqueror).

KDE se compone de subsistemas que realizan la mayor parte del trabajo. Los subsistemas más importantes son:

DCOP	Desktop Communication Protocol.
KIO	Network Transparent I/O.
SYCOCA	System Configuration Cache.
KParts	Componentes integrados
KHTML	librería HTML 4.0
XMLGUI	Arquitectura dinámica GUI, basad en XML.
aRts	Sistema Multimedia.

GNOME permite el empleo de cualquier gestor de ventana que se ajuste a sus especificaciones, en la actualidad algunos gestores más conocidos son: Sawfish, Enlightenment, Window Maker, IceWM, Scwm y FVWM (versión 2.3). El Administrador de ficheros de GNOME se denomina Nautilus. De toda la lista de paquetes disponibles para GNOME, algunos son parte integrante del propio entorno y otros sólo se necesitan si se va a desarrollar haciendo uso de sus funcionalidades. Algunos de los paquetes de GNOME son:

ORBit	es el proveedor de CORBA, es el que permite la comunicación entre las diferentes partes de GNOME.
GTK	es la Gimp ToolKit (GIMP viene de GNU Image Manipulation Program que es una aplicación de manipulación de gráficos de código abierto). Permite la creación de botones, barras de desplazamiento y otros elementos básicos.
Imlib	es una librería de dibujo (render), es la responsable de indicar a X-Windows como dibujar las cosas y como cargar imágenes.
gnome-libs	contiene las librerías básicas para todas las aplicaciones GNOME.
glib	contiene rutinas que emplean casi todos los programas de GNOME.
libxml	permite interpretar los archivos en XML, a veces se conoce como gnome-xml.
libglade	librería que lee ficheros XML, mediante libxml y lo convierte en interfaz de usuario.
libgtop	sirve para dar a GNOME información sobre la organización de los sistemas de archivo y como el sistema controla lo que se ejecuta y cuando.
libghttp	la librería HTTP de LINUX.
esound server	es el encargado de manejar el sonido bajo LINUX.
gnome-print	ofrece un entorno homogéneo de impresión a las aplicaciones.
bug-buddy	programa para ayudar en la comunicación de errores (bugs).
gdk-pixbuf	es una librería para la carga de imágenes, dibujo y carga de animaciones simples.
eog	es un visor de imágenes pequeño y rápido.
bonobo	permite implantar contenidos de un programa dentro de otro (embeber).
oaf	(Object Activation Framework), se emplea en las nuevas aplicaciones GNOME para buscar y ejecutar objetos CORBA.
gconf	es la aplicación encargada de mantener las opciones de configuración.

### 6.3. LA INTERFAZ HOMBRE-MÁQUINA DE WINDOWS.

Una característica muy importante en Windows 2000 es la incorporación de los servicios de terminal, lo que permite que un sistema sea multiusuario. Eso ha obligado a Windows 2000 a una reconstrucción del subsistema win32 para llevar control de los diferentes usuarios del sistema conectados en un momento dado.

Todos los sistemas operativos de la familia Windows, son de tipo GUI (Graphics User Interface). Sus elementos principales son las ventanas gráficas, los menús, el teclado y el dispositivo apuntador, usualmente un ratón. En el caso de Windows, la interfaz gráfica constituye el mayor empeño y complejidad de todo el sistema operativo, siendo parte integrante de él.

Cuando un usuario inicia una sesión en una máquina Windows, se le presenta una pantalla denominada Escritorio. Conteniendo una barra de tareas, sobre la que se encuentra un botón de Inicio des-



de el que se despliega todo el sistema de menús. Sobre el tapiz del escritorio se encuentran una serie de iconos, que constituyen los accesos directos a los diferentes programas instalados en el sistema.

El acceso directo es un tipo de archivo, de pequeño tamaño, que contiene toda la información necesaria para acceder a un programa o aplicación determinada, ocultando al usuario la complejidad de conocer su ubicación exacta en el sistema de almacenamiento o los parámetros necesarios para su ejecución. Un acceso directo puede referirse también a un directorio, constituyendo entonces un atajo dentro de la jerarquía del sistema de archivos.

La gestión del sistema de archivos se realiza a través del explorador de Windows, programa que permite la navegación a través de la jerarquía de directorios, tanto de la máquina, como de otros directorios compartidos a través de la Red.

A diferencia de UNIX o LINUX, la arquitectura GUI de Windows es un bloque indivisible del propio sistema operativo, no es divisible en subsistemas o paquetes, ni es reemplazable, aunque sí permite su ampliación mediante la incorporación de componentes de terceros, que empleen la tecnología COM (Common Object Model), propia de Microsoft.

## 7. ESTADO ACTUAL DE LOS S.O.

### 7.1. UNIX, LINUX.

En la actualidad, existen registradas 353 distribuciones diferentes de LINUX, que abarcan un amplio abanico de plataformas hardware.

Existen distribuciones de LINUX para plataformas Intel, PPC, Alpha, Sparc, Itanium, para Mainframe, para procesadores Motorola 68000, para arquitecturas de 64bit y para sistemas empotrados o embebidos, como teléfonos móviles, y sistemas en tiempo real (la nasa ha desarrollado su propia distribución para las naves espaciales).

Por su parte UNIX, se mantiene en plataformas más específicas, soportadas en gran medida por los respectivos fabricantes de hardware, si bien existen distribuciones libres como FreeBSD, OpenBSD y de software abierto como NetBSD.

Entre los sistemas UNIX propietarios, se encuentran AIX 5L de IBM, HP-UX 11iv2 de HP, Solaris 10 de SUN microsystems.

### 7.2. WINDOWS.

La familia de sistemas operativo Windows tiene en la actualidad tres variantes principales:

- Sistemas operativos para servidores.
- Sistemas operativos para estaciones de trabajo y ordenadores personales.
- Sistemas operativos para equipos portátiles.

## Sistemas Operativos de Servidor.

Los sistemas operativos de servidor están pensados para una amplia gama de sistemas, desde pequeños servidores departamentales hasta grandes sistemas multiprocesadores corporativos.

La última versión de estos sistemas es Windows Server 2003, disponible en diferentes variantes o «ediciones», adecuadas a diferentes tamaños del entorno de trabajo:

- Edición Estándar (Standar Edition).
- Edición Empresarial (Enterprise Edition).
- Edición de Centro de Datos (DataCenter Edition).
- Edición Web (Web Edition).
- Servidor de pequeñas empresas (Small Business Server).

### *Edición Estándar.*

Diseñada para cargas normales de trabajo a nivel departamental, soporta multiproceso simétrico hasta 4 procesadores y 4GB de memoria RAM.

### *Edición Empresarial.*

Soporta servidores de alto rendimiento y la capacidad de creación de clusters de servidores. Soporta multiproceso simétrico hasta 8 procesadores, 32 GB de memoria RAM y clúster de 8 nodos. En su versión de 64 bits, para el procesador intel Itanium, soporta hasta 8 procesadores, 64 GB de RAM y clúster de 8 nodos.

### *Edición Centro de Datos.*

Diseñado para el mayor nivel de fiabilidad y escalabilidad disponible. Soporta soluciones de «misión crítica» para bases de datos y elevado volumen de proceso de transacciones en tiempo real. Soporta multiproceso simétrico hasta 32 procesadores y 64 GB de memoria RAM. En su versión de 64 bits, para el intel Itanium, soporta hasta 64 procesadores, 512 GB de memoria RAM y clúster de 8 nodos.

### *Edición Web.*

Diseñada fundamentalmente como servidor web y de alojamiento de webs de terceros (Hosting), se ofrece como una solución económica y efectiva para organizaciones con necesidad de distribuir rápidamente páginas, servicios, aplicaciones y sitios web.

Presenta, como característica particular, la prohibición de instalar y ejecutar sobre ella aplicaciones que no sean de servidor web, pues su uso está restringido en exclusiva a este entorno.

### *Servidor de pequeñas empresas.*

No es una versión particular del sistema operativo, sino una distribución de software especializada que, junto a Windows Server 2003 como sistema operativo, incorpora elementos como mensajería (Exchange Server 2003), Bases de Datos (Microsoft SQL Server 2003) y acceso a Internet corporativo (ISA Server), ofreciendo en un solo producto todo lo necesario para implantar un sistema de información en pequeñas empresas.

### **Sistemas Operativos para Ordenadores Personales.**

Para este tipo de plataformas, Microsoft dispone de la familia de sistemas operativos Windows XP también disponible en diferentes variantes o ediciones:

- Edición Home.
- Edición Profesional.
- Edición Media Center.
- Edición Tablet PC.
- Edición 64 bits.

#### *Edición Home.*

Esta es la versión básica de Windows XP. Está pensada para usuarios domésticos con pocas necesidades de conectividad y bajos requerimientos en seguridad y privacidad (equipos independientes).

#### *Edición Profesional.*

Esta es la edición usual y más difundida de Windows XP. Sobre la edición Home, soporta la conexión a grandes redes, con mejoras en la seguridad y la privacidad.

#### *Edición Media Center.*

Orientada a usuarios multimedia, esta edición ofrece soporte para la gestión de los dispositivos multimedia de un hogar (Televisión, música, DVDs, etc.) se soporta en una plataforma hardware específica denominada Media Center PC que, entre otros elementos incluye:

- Un Control Remoto compatible con Media Center
- Un Sensor de Infrarrojos.
- Una tarjeta gráfica avanzada.
- Un sintonizador de televisión.
- Un codificador hardware.

- Una salida de televisión.
- Una salida digital de audio.

#### *Edición Tablet PC.*

La edición Tablet PC va dirigida a las plataformas Tablet PC de los diversos fabricantes, ofreciendo la funcionalidad de Windows XP Profesional más las utilidades y herramientas adecuadas a la operación de este tipo de equipos (escritura en pantalla, etc.).

#### *Edición 64 bits.*

Es el sistema operativo para las plataformas Itanium de Intel. Soporta hasta 16 GB de memoria RAM y 16 TeraBytes de memoria virtual. Permite multiproceso simétrico con 2 procesadores.

### **Sistemas Operativos para equipos Portátiles.**

Para los sistemas portátiles, microsoft dispone de dos familias de productos:

- Windows Incorporado (Embebido o incrustado -Embedded-).
- Windows Mobile.

#### *Windows Incorporado.*

Son sistemas operativos que se incorporan al producto en el proceso de fabricación del mismo, siendo inseparables de él. Se emplean para aplicaciones diversas, como el mercado de los microcontroladores para procesos industriales en tiempo real, automoción, electrodomésticos, domótica, etc.

En esta familia de sistemas operativos se encuentra el Microsoft Windows CE, el Windows CE.Net y el más novedoso Windows XP Embedded.

#### *Windows Mobile.*

Sistema operativo para el entorno de equipos portátiles orientados a la comunicación, como los terminales multifunción de telefonía (el smartphone es un teléfono multimedia y multifunción), los equipos Pocket PC o los Portable Media Center que son equipos pensados para acceder, desde la palma de la mano, a todos los contenidos multimedia disponibles en un ordenador Media Center PC, ejecutando Windows XP Media Center Edition.

## BIBLIOGRAFÍA

- *UNIX shell differences and how to change your shell*. BRIAN BLACKMORE.
- *Sistemas Operativos Modernos*. ANDREW S. TANENBAUM. Ed. Prentice Hall
- Apuntes del Curso Selectivo del Cuerpo Superior de Sistemas y Tecnologías de la Información de la Administración del Estado. Instituto Nacional de Administración Pública.
- Free Software Foundation. <http://www.fsf.org>.
- Open Source Initiative. <http://www.opensource.org>.
- Linux Online. <http://www.linux.org>.
- Operating System Technical Comparison. <http://www.osdata.com>.
- MSDN Microsoft Developer Network. <http://msdn.microsoft.com/library/default.asp>.
- IBM Library Server®, Library. <http://publibfp.boulder.ibm.com/cgi-bin/bookmgr/library>.
- Apuntes del departamento de ingeniería de computadoras. Universidad de Tübingen.
- Temario de las pruebas selectivas para ingreso en el Cuerpo Superior de Sistemas y Tecnologías de la Información de la Administración del Estado. Astic.
- Temario de las pruebas selectivas para el acceso, por promoción interna, al Cuerpo de Gestión de Sistemas e Informática de la Administración del Estado. Ministerio para las Administraciones Públicas.



