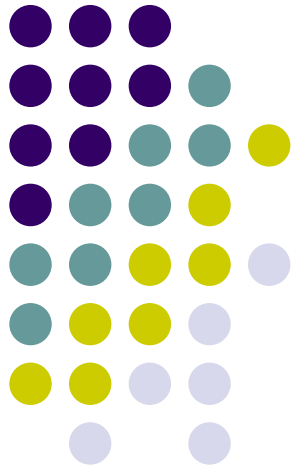
Tema 3. Sistemas Operativos



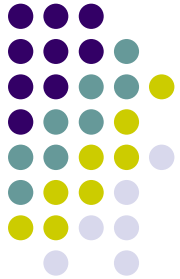
Jesús María Aransay Azofra

Informática

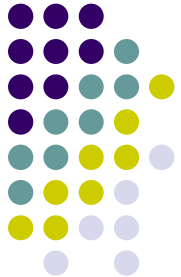
Universidad de La Rioja 2011/2012

Índice

3.1 Misión y necesidad del SO



1. Qué entendemos por sistema operativo
2. Qué tipo de tareas nos evita el sistema operativo (programación de bajo nivel, ensamblador)
3. Qué facilidades nos suele aportar un sistema operativo
4. Estructura y tipos de sistemas operativos
5. Llamadas al sistema

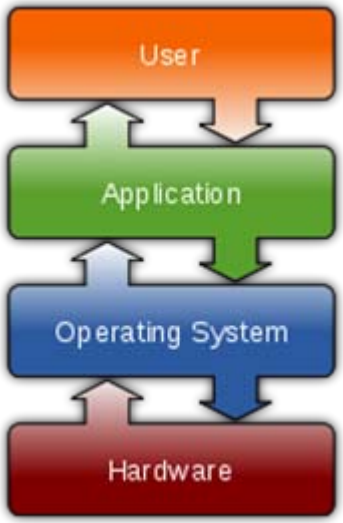


3.1.1 Definición de SO

Existen muy diversas definiciones de Sistema Operativo. Sin embargo, todas ellas tienen características comunes, que nos servirán para crear una idea concreta de qué es o para qué sirve un SO

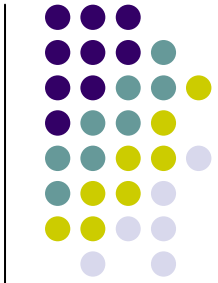
3.1.1 Definición de SO

- An **operating system** (**OS**) is software, consisting of programs and data, that runs on computers and manages the computer hardware and provides common services for efficient execution of various application software:



http://en.wikipedia.org/wiki/Operating_system

3.1.1 Definición de SO



- The operating system (OS) is specialized computer software that allocates memory and manages system resources. When a computer is turned on, the OS is loaded into memory and works as an abstraction layer between the physical hardware and the software. While the OS doesn't perform a specific function it helps other programs run smoothly and efficiently.

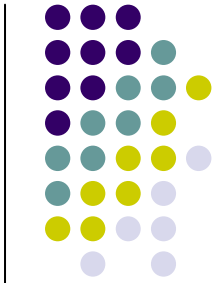
http://en.wikiversity.org/wiki/Operating_system

3.1.1 Definición de SO

- "… the operating system is like a (orchestra's) conductor: It is responsible for coordinating all of the computer's individual components so that they work according to a single plan.…. Similarly, the OS allocates the computer's components to different programs, synchronizes their individual activities, and provides the mechanisms that are needed so that the programs execute in perfect harmony."

"Operating Systems". Gary Nutt (Addison Wesley)

3.1.1 Definición de SO

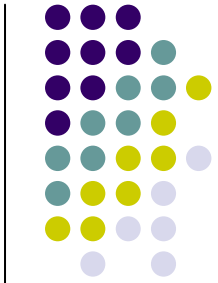


Un SO es un programa que tiene encomendadas una serie de funciones cuyo objetivo es simplificar el manejo y la utilización del computador, haciéndolo seguro y eficiente. Las funciones clásicas del SO se pueden resumir en:

- Gestión de los recursos del computador
- Ejecución de servicios para los programas en ejecución
- Ejecución de los mandatos de los usuarios

"Sistemas operativos. Una visión aplicada".
J. Carretero Pérez *et al*. McGraw Hill

3.1.1 Definición de SO

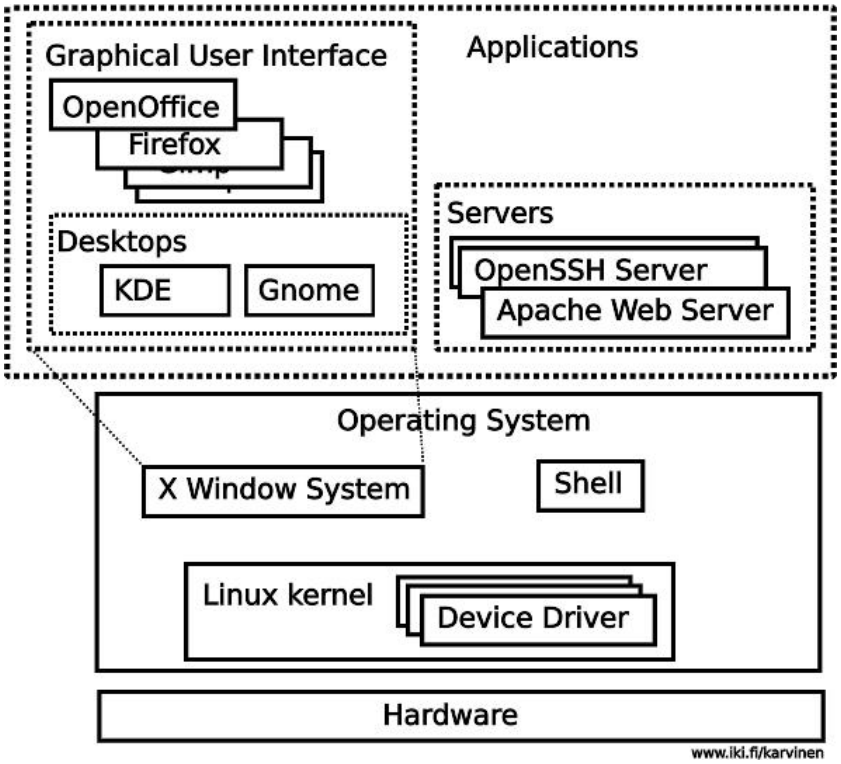


Tratando de resumir, un SO es:

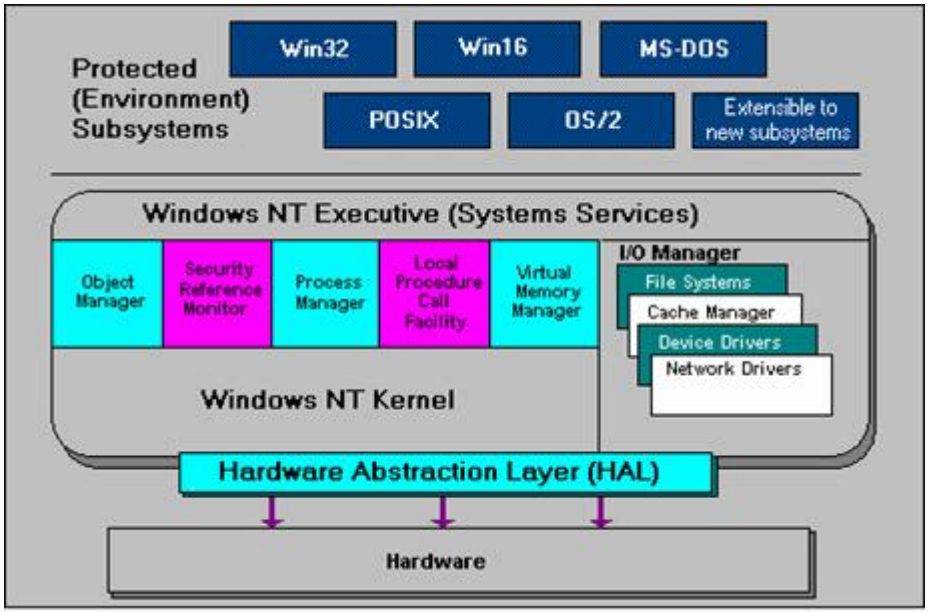
- Un conjunto de componentes de software (de bajo nivel o de sistema) sobre las que funciona el software (de alto nivel o de aplicaciones)
- Un gestor y coordinador de los recursos del ordenador
- Un capa de abstracción entre la parte física (hardware) y las aplicaciones (software)

3.1.1 Definición de SO

Arquitectura de un sistema Linux



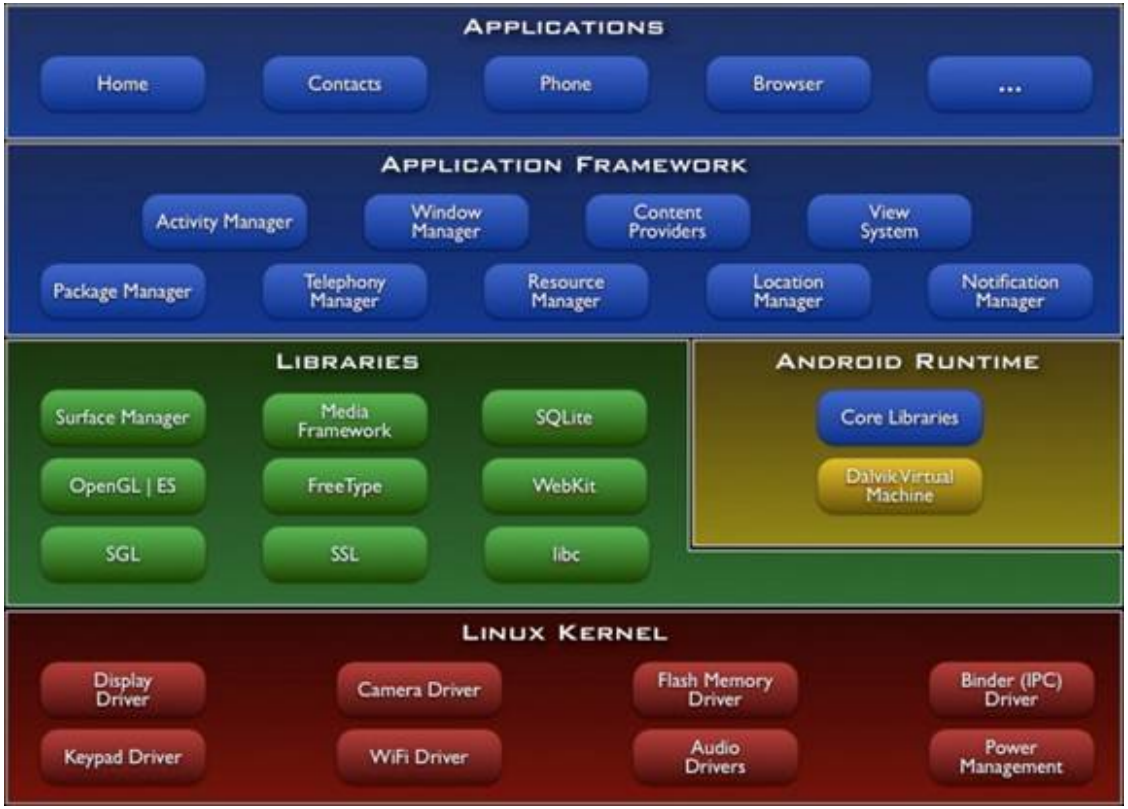
Arquitectura de sistema Windows NT



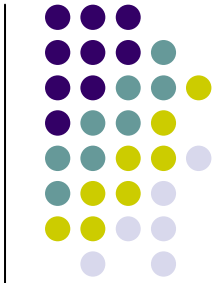
¿Dónde reside el Sistema Operativo en ambos casos?

3.1.1 Definición de SO

Arquitectura de un sistema Android, sistema basado en Linux (no aparece el hardware)



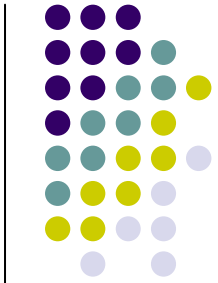
3.1.2 Qué tareas nos evita un SO



Algunos ejemplos que ya hemos visto:

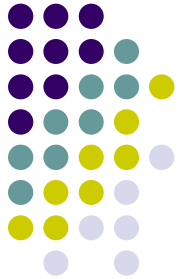
- Cuando guardamos un fichero en disco duro, la localización en memoria no se hace por “Cilindros – Cabeza - Sector”, sino por ficheros (sistemas de ficheros fat, fat32, ntfs, ext)
- Los ficheros de texto no se deben “escribir” en binario (unos y ceros), sino usando caracteres de texto conforme a una tabla de caracteres (utf8, iso-8859-…)
- La navegación por Internet no se debe hacer dividiendo la información en paquetes y asignándoles puertos, IP’s, direcciones MAC; las aplicaciones (navegador…) y el SO nos evitan esa tarea
- Al conectar un USB (un periférico) a nuestro ordenador no debemos buscar un controlador para el mismo, “montarlo”… El SO gestiona los dispositivos de I/O

3.1.2 Ejemplos de tareas que nos evita un SO



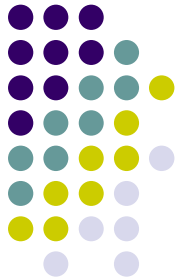
- Nos evita realizar programación de bajo nivel (lenguaje ensamblador). El SO dota de APIs (interfaces o librerías de programación de aplicaciones) que abstraen la configuración de la máquina. Por ejemplo, “Win32 API”, “Win16 API” en Windows, “POSIX” (Portable Operating System Interface) para UNIX, Linux y WinNT

3.1.2 Ejemplos de tareas que nos evita un SO



- Coordinación y priorización de tareas (o procesos), gestión de recursos. El sistema operativo es capaz de ejecutar distintos procesos (reproducir música, enviar un mail, mantener la interfaz gráfica…) de forma que todos ellos se ejecuten simultáneamente (bien porque se paralelizan, bien porque se secuencian de modo que el usuario no lo nota)

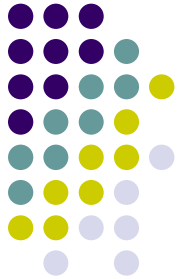
3.1.2 Ejemplos de tareas que nos evita un SO



Aplicaciones | Procesos | Servicios | Rendimiento | Funciones de red | Usuarios

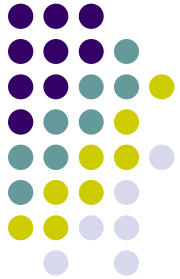
Nombre de imagen	Nombre de usuario	CPU	Memoria ...	Descripción
chrome.exe	jearansa	00	74.168 KB	Google Chrome
chrome.exe	jearansa	00	55.152 KB	Google Chrome
chrome.exe	jearansa	00	42.260 KB	Google Chrome
explorer.exe	jearansa	00	37.280 KB	Explorador de Windows
chrome.exe	jearansa	00	34.832 KB	Google Chrome
chrome.exe	jearansa	00	30.964 KB	Google Chrome
devcpp.exe	jearansa	00	30.716 KB	Dev-C++ IDE
dwm.exe	jearansa	00	26.036 KB	Administrador de ventanas del escritorio
WINWORD.EXE	jearansa	00	25.428 KB	Microsoft Office Word
POWERPNT.EXE	jearansa	01	22.476 KB	Microsoft Office PowerPoint
chrome.exe	jearansa	00	22.212 KB	Google Chrome
chrome.exe	jearansa	00	19.916 KB	Google Chrome
HPPA_Main.exe	jearansa	00	16.752 KB	HP Power Assistant
PrivacyIconClient.exe	jearansa	00	12.728 KB	Intel(R) Management and Security Status
HPWA_Main.exe	jearansa	00	12.664 KB	HP Wireless Assistant
chrome.exe	jearansa	00	11.220 KB	Google Chrome
chrome.exe	jearansa	00	9.192 KB	Google Chrome
csrss.exe		01	6.620 KB	
sttray.exe	jearansa	00	5.992 KB	IDT PC Audio
BTTray.exe	jearansa	00	3.524 KB	Bluetooth Tray Application
SynTPEnh.exe	jearansa	00	3.120 KB	Synaptics TouchPad Enhancements
LightScribeControlPanel.exe	jearansa	00	3.096 KB	LightScribeControlPanel.exe
SnippingTool.exe	jearansa	00	3.088 KB	Recortes

3.1.3 Facilidades que aporta un SO



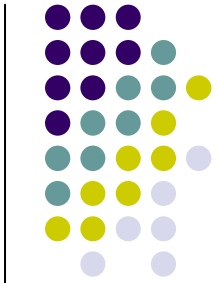
- Es difícil definir una lista cerrada y completa de facilidades que aporta un sistema operativo, depende del SO y de su finalidad. En la siguiente lista no pretendemos ser exhaustivos, aunque deberíamos ser capaces de señalar las principales funciones que aporta cualquier (o al menos algunos) SO

3.1.3 Facilidades que aporta un SO



- Interfaz de usuario
  - Interfaz de línea de mandatos
  - Interfaz gráfica de usuario
- El kernel (o núcleo)
  - Ejecución de programas
  - Interrupciones
  - Modos
  - Gestión de memoria
  - Memoria Virtual
  - Multitarea
  - Acceso a discos y sistema de ficheros
  - Drivers (controladores) de dispositivos
- Funciones de red
- Seguridad

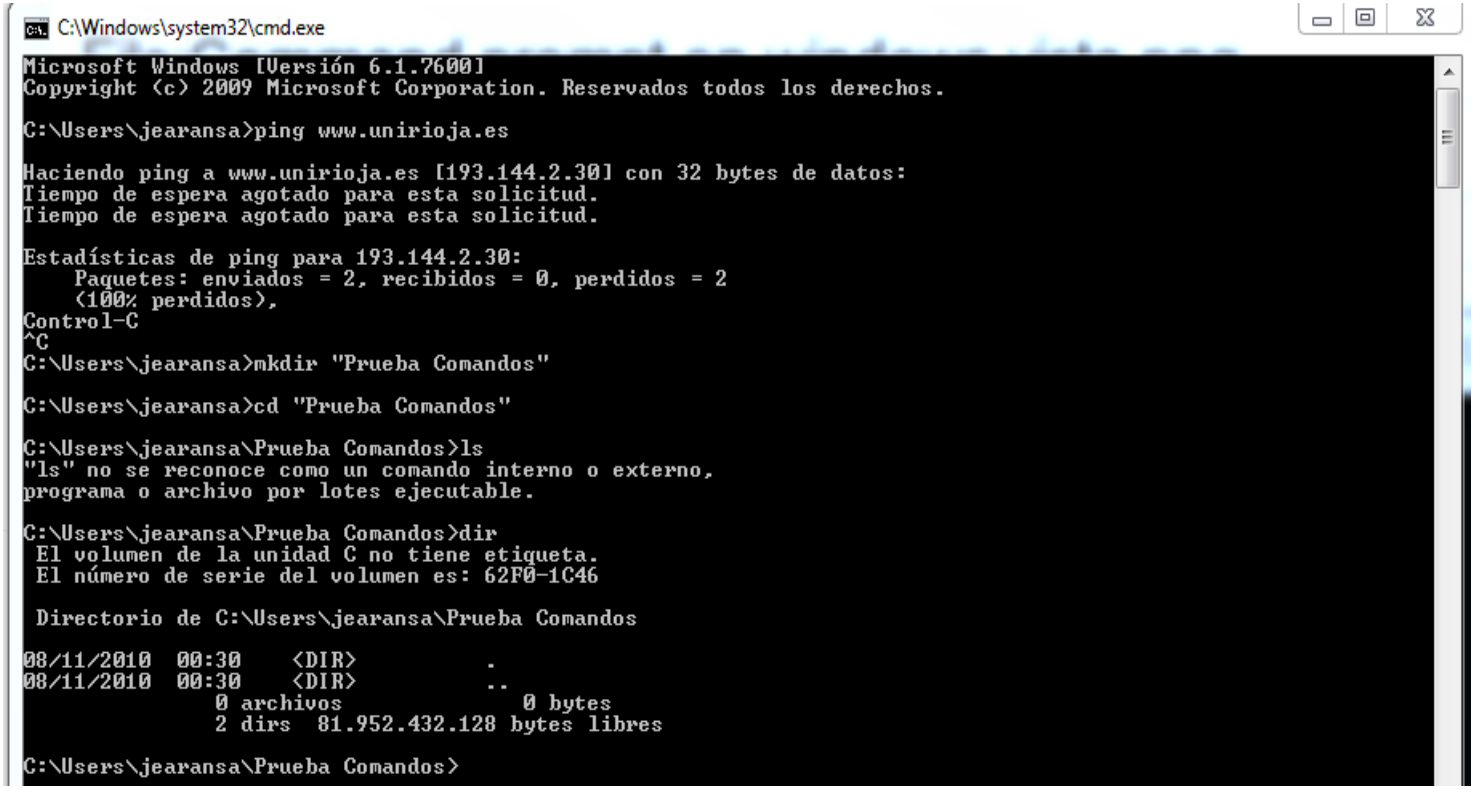
3.1.3 Interfaz de usuario



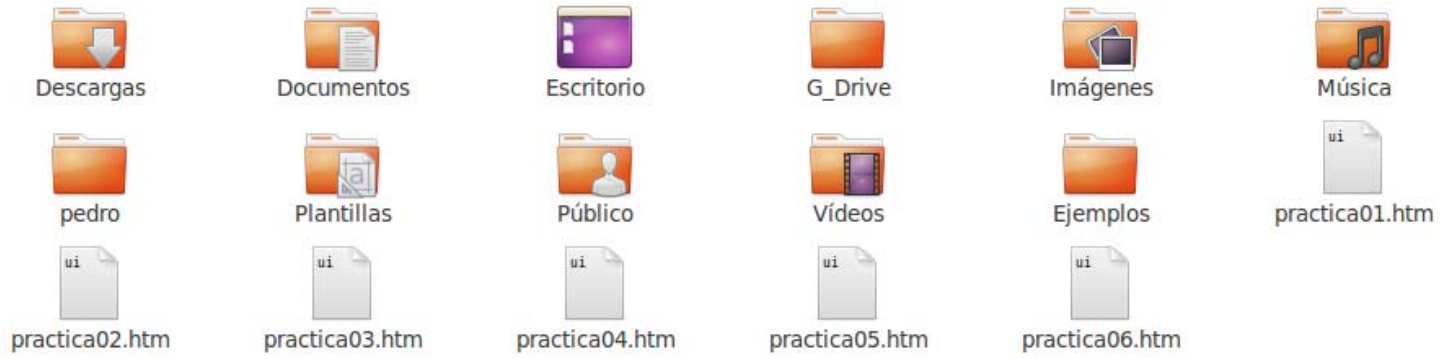
- Cualquier dispositivo que interactúa con seres humanos requiere una interfaz de usuario. En los SO generalmente son de dos tipos:
  - Interfaz de línea de mandatos (Command Line Interface, CLI)
  - Interfaz gráfica de usuario (Graphical User Interface, GUI)

3.1.3 Interfaz de línea de mandatos

Es un mecanismo para interactuar con el sistema operativo o con el software de un ordenador por medio de la escritura de comandos que llevan a cabo tareas específicas



3.1.3 Interfaz de línea de mandatos



Para poder borrar todos los ficheros de nombre practica__.___ desde la línea de mandatos

```
jesus@jesus-laptop:~$ rm practica*.*
```

Para algunas tareas repetitivas, es más rápido (y por supuesto consume menos recursos) que la interfaz gráfica

3.1.3 Interfaz gráfica de usuario (GUI)

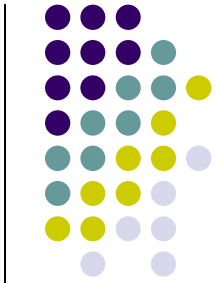
- Tipo de interfaz de usuario que permite interactuar no sólo escribiendo mandatos (interfaces para ordenadores, reproductores de música o vídeo…), sino por medio de otros dispositivos, como ratón, táctiles…

Suelen estar basadas en el uso de iconos (para representar carpetas, ficheros, programas)

3.1.3 Interfaz gráfica de usuario (GUI)

Su gran ventaja es la facilidad de uso y la rapidez de aprendizaje, aparte de la mejor apariencia visual

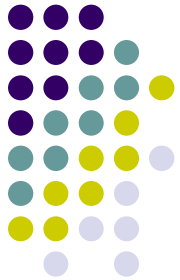
3.1.3 Kernel



- Es la parte central de los SO. Constituye el puente entre el software de aplicaciones y el procesado de datos al nivel de hardware. También se encarga de gestionar procesos, uso de memoria, acceder al sistema de ficheros…

Veamos algunas de esas tareas de forma más detallada

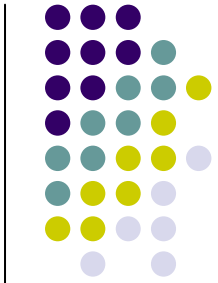
3.1.3 Kernel: ejecución de programas



- Cada vez que se ejecuta un programa en el ordenador, el kernel crea uno o varios procesos, asigna memoria y otros recursos, establece una prioridad para el proceso, carga el código del programa a memoria principal y finalmente ejecuta el programa

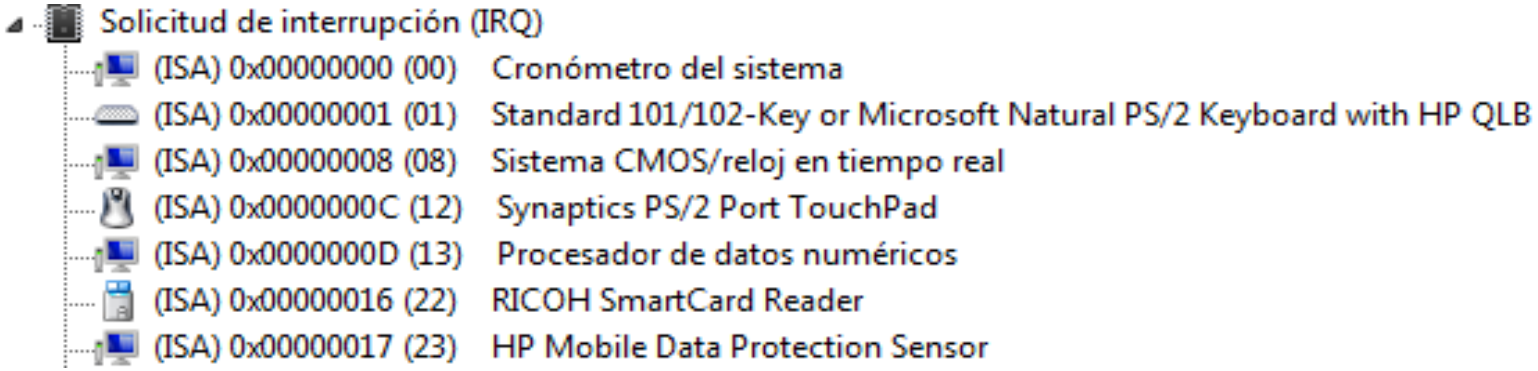
Nombre de imagen	CPU	Espacio de trabajo (memoria)	Espacio máximo de trabajo (memoria)	Tamaño de asignación	Línea de comandos	Descripción
VirtualBox.exe	02	84.064 KB	94.128 KB	121.032 KB	C:\PROGRA~1\Oracle\VIRTUA~1\Virtu...	VirtualBox GUI
chrome.exe	00	85.800 KB	129.200 KB	66.380 KB	"C:\Users\jearansa\AppData\Local\Go...	Google Chrome
chrome.exe	00	58.696 KB	77.044 KB	49.552 KB	"C:\Users\jearansa\AppData\Local\Go...	Google Chrome
explorer.exe	01	62.776 KB	77.300 KB	46.796 KB	C:\Windows\Explorer.EXE	Explorador de Windows
chrome.exe	00	42.500 KB	73.864 KB	43.924 KB	"C:\Users\jearansa\AppData\Local\Go...	Google Chrome
WINWORD.EXE	00	45.564 KB	55.064 KB	40.592 KB	"C:\Program Files\Microsoft Office\OFF...	Microsoft Office Word

3.1.3 Kernel: gestión de interrupciones

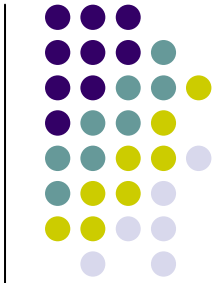


- Las distintas componentes del ordenador (CPU, memoria RAM, disco duro, tarjeta de red) funcionan a distintas velocidades; su coordinación requiere hacer esperar a unos dispositivos, o priorizar a otros. Esto se consigue por medio de interrupciones
- Una interrupción es una señal por la que el software o hardware del ordenador avisa al SO de que ha completado una tarea (p. ej. leer un fichero) para que éste pueda llevar a cabo su siguiente acción (p. ej. copiarlo a un USB)
- Antes de que existieran las interrupciones el SO debía comprobar si una tarea estaba completa (de forma cíclica), para poder comenzar la siguiente. La pérdida de tiempo y recursos era considerable

3.1.3 Kernel: Interrupciones



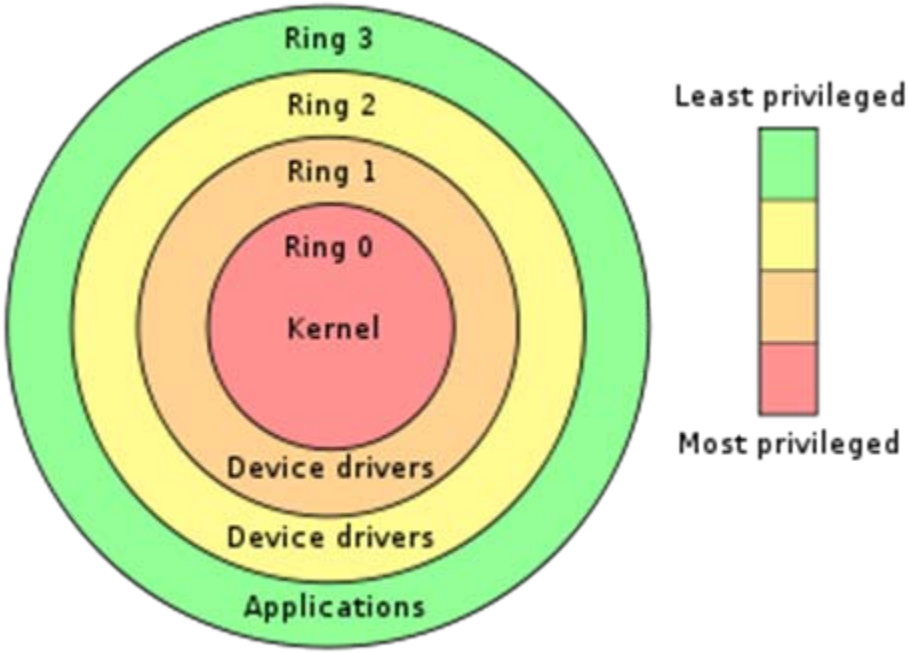
3.1.3 Kernel: modos



El kernel gestiona el modo supervisor y el modo protegido. Algunos SO sencillos (MSDOS) solo incluyen modo supervisor. Otros más complejos (Linux, Windows) incluyen ambos:

- Modo supervisor: usado para tareas de bajo nivel que requieren acceso ilimitado al hardware (lectura y escritura directa a memoria, comunicación con la tarjeta gráfica). La BIOS, por ejemplo, se ejecuta en modo supervisor; el SO, en general, también
- Modo protegido: el kernel, cuando inicia un proceso, puede decidir que el mismo se ejecute en modo protegido, prohibiendo el acceso del mismo a las órdenes de la CPU

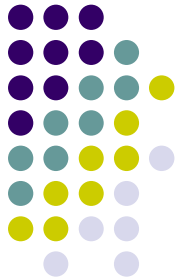
3.1.3 Kernel: modos



Los procesadores x86 incluyen cuatro posibles modos (o anillos), dependiendo del nivel de privilegios de que se dispone en cada uno de ellos:

- Ring 0: privilegios totales, acceso a la CPU o la RAM
- Ring 1: Acceso al hardware (cámaras, ratones, teclados)
- …
- Ring 3: ejecutar programas de usuario…

3.1.3 Kernel: gestión de memoria

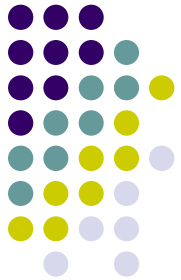


- El kernel debe asegurar que la memoria principal (o RAM) en uso por cada programa no sea usada simultáneamente por ningún nuevo programa
- Modo de funcionamiento (en breve): la memoria principal se segmenta en fragmentos, cada uno de lo cuales es accesible sólo por un programa. Si un programa trata de acceder a un fragmento que no es el suyo, se producirá una “violación de segmentación”, y el kernel terminará el programa “invasor”

3.1.3 Kernel: gestión de memoria

Nombre de imagen	Espacio de trabajo (memoria)	Espacio máximo de trabajo (memoria)
VirtualBox.exe	84.072 KB	94.128 KB
chrome.exe	88.792 KB	129.200 KB
chrome.exe	66.832 KB	78.540 KB
explorer.exe	64.928 KB	77.300 KB
chrome.exe	42.804 KB	47.424 KB
chrome.exe	42.500 KB	73.864 KB
WINWORD.EXE	45.936 KB	55.064 KB
devcpp.exe	35.332 KB	46.112 KB
chrome.exe	33.628 KB	46.696 KB
HPPA_Main.exe	39.076 KB	44.328 KB
dwm.exe	32.124 KB	87.276 KB
VirtualBox.exe	25.368 KB	25.368 KB
PrivacyIconClient.exe	19.732 KB	56.240 KB
HPWA_Main.exe	30.816 KB	36.020 KB
chrome.exe	22.180 KB	32.432 KB
csrss.exe	34.992 KB	88.804 KB
sttray.exe	12.124 KB	14.476 KB
VBoxSVC.exe	9.364 KB	9.384 KB
POWERPNT.EXE	5.968 KB	50.276 KB

3.1.3 Kernel: memoria virtual

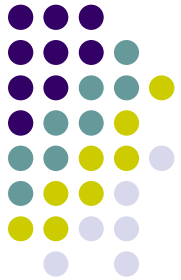


Además de segmentar la memoria, el kernel puede elegir zonas de memoria para ser usadas por varios programas (pero no simultáneamente, recordar la gestión de memoria)

La memoria se “pagina”, y estas páginas, accesibles para diversos procesos, se conocen como “memoria virtual”

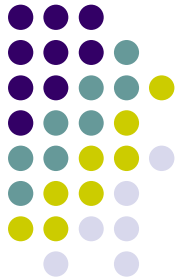
Si un proceso trata de acceder a una página de memoria virtual usada por otro proceso se produce una interrupción y el kernel, posiblemente, le asignará una nueva página

3.1.3 Kernel: multitarea



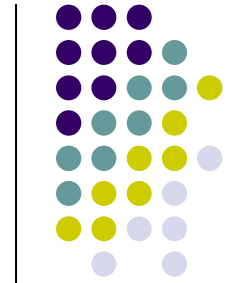
- En los ordenadores con una CPU, sólo una tarea o proceso se puede estar ejecutando en un momento (en cada ciclo de reloj o Hertzio; de ahí los procesadores de “2,4GHz”…)
- La multitarea, gestionada por el kernel por medio de un “scheduler” (planificador), da apariencia de que varios procesos se ejecutan a la vez
- El planificador determina cuánto tiempo va a pasar ejecutándose cada programa y da acceso a cada uno (secuencialmente) para usar la CPU y la memoria

3.1.3 Kernel: acceso a discos y sistema de ficheros



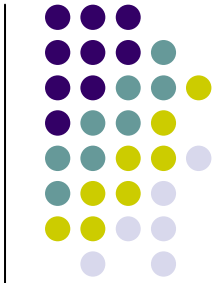
- El acceso a datos guardados en dispositivos de entrada o salida (discos duros, usb's, discos externos) es una de las misiones centrales de los SO
- Los ficheros permiten acceso más rápido (recordad las tablas FAT), mayor fiabilidad, y mejor uso del espacio libre
- Los ficheros, además, permiten otras propiedades como asignar nombres, programas de apertura, compresión, permisos para usuarios…
- A esta forma de organización se la conoce como "sistema de ficheros", y suele estar basada en un árbol de directorios

3.1.3 Kernel: acceso a discos y sistema de ficheros



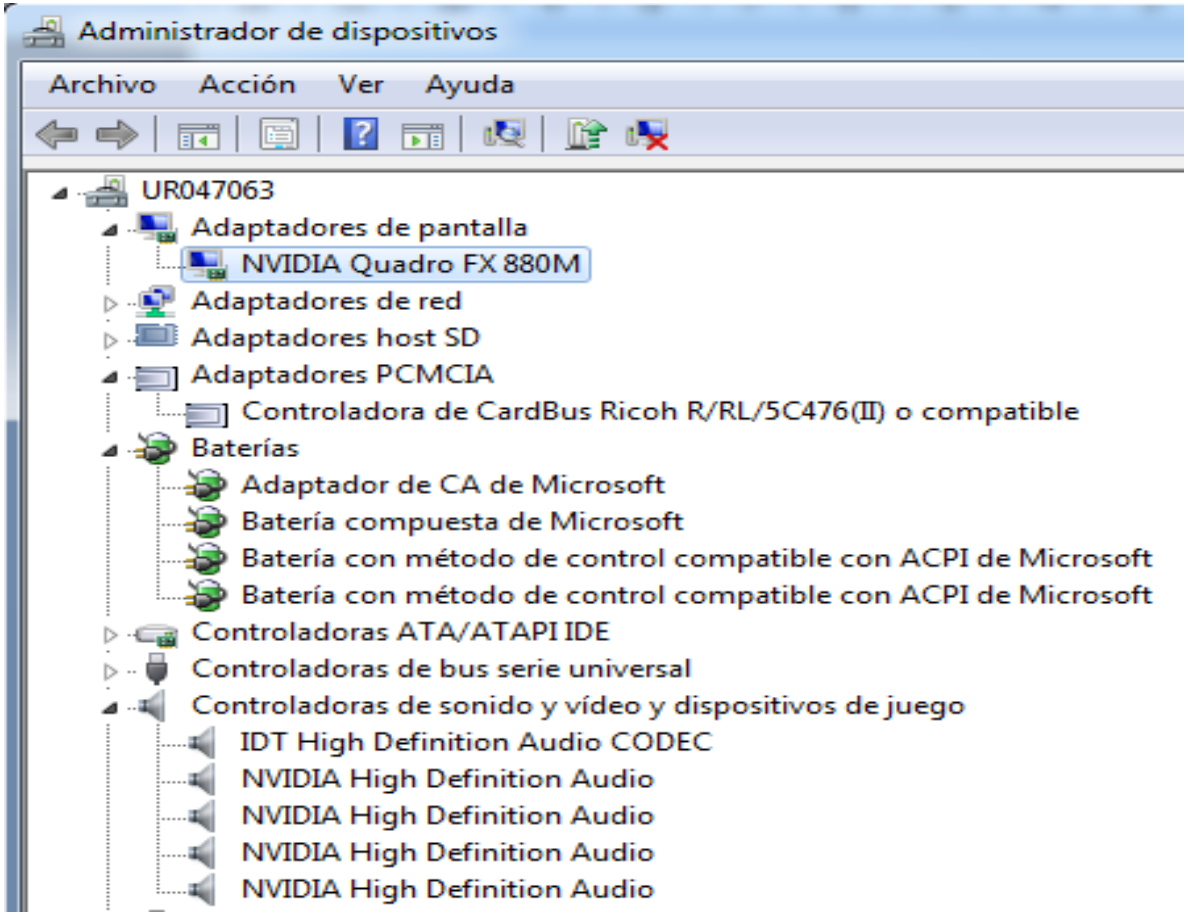

```
drwx------  3 jesus jesus 4096 2010-11-05 11:41 .thumbnails
drwx------  2 jesus jesus 4096 2010-11-04 17:25 .update-notifier
-rw-r-----  1 jesus jesus    5 2010-11-08 00:48 .vboxclient-clipboard.pid
-rw-r-----  1 jesus jesus    5 2010-11-08 00:48 .vboxclient-display.pid
-rw-r-----  1 jesus jesus    5 2010-11-08 00:48 .vboxclient-seamless.pid
drwxr-xr-x  2 jesus jesus 4096 2010-11-04 17:24 Vídeos
-rw-------  1 jesus jesus 2608 2010-11-08 02:59 .xsession-errors
-rw-------  1 jesus jesus 4163 2010-11-05 14:55 .xsession-errors.old
```

3.1.3 Kernel: controladores de dispositivos



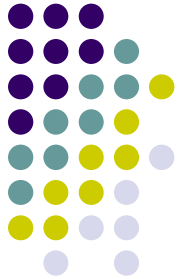
- Los controladores de dispositivos (drivers) son programas que permiten la interacción con determinado hardware
- Permiten la interacción del kernel con los dispositivos, y proveen del sistema de interrupciones necesario para la comunicación

3.1.3 Kernel: controladores de dispositivos



Los dispositivos tienen su controlador para que el kernel pueda comunicarse con ellos. Generalmente, cada “tipo” de dispositivos tiene un driver genérico que se debe aplicar a cada dispositivo

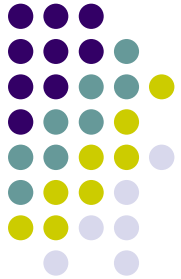
3.1.3 Kernel: controlador de dispositivos



```
/*-----------------------------------------
 *  MAIN
 *-----------------------------------------*/
int
main (int argc, char *argv[])
{
  GtkWidget     *window1;
  GtkWidget     *print_continue;
  GtkWidget     *print_cancel;
  struct    sockaddr_in address;
  struct    hostent *host;
  struct    sigaction   sa;

  int       ret,    retry;
  char      *model_s,model_name[30];

#ifdef DEBUG
  create_log();
#endif

  bindtextdomain (PACKAGE, PACKAGE_LOCALE_DIR);
  textdomain (PACKAGE);

  /*-------------------------
     Initialize
  -------------------------*/
  SetGtkResourceDefault();

  gtk_set_locale ();
  gtk_init (&argc, &argv);

  if( (model_s = check_arg( argc,argv )) != NULL )
    if( (model_id = check_model_s( model_s )) >= MODEL_CNT)
        model_s = NULL;
  flag_noparent = check_noparent( argc, argv );
```

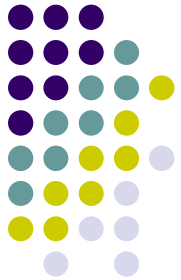
Aspecto de un driver de Linux para un modelo concreto de impresora: en este caso, es un programa (conjunto de librerías) en C (o C++) que hace llamadas al sistema

3.1.3 Funciones de red



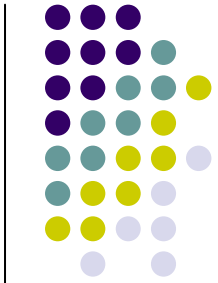
- La mayor parte de los SO soportan una variedad de protocolos de red, hardware, y aplicaciones para usarlos (Tema 2)
- Por medio de funciones de red el SO puede acceder a recursos de ordenadores remotos y usarlos como si fueran locales (práctica 8)
- A través de funciones del SO “configuramos” los distintos parámetros de nuestra red (IP, máscara de red, servidores DNS…)

3.1.3 Seguridad



- El SO, estando conectado a Internet o redes locales, debe ser capaz de distinguir las peticiones que recibe que deben ser procesadas y las que no; por ejemplo, por medio del uso de usuarios y contraseñas, del uso de modos de ejecución (privilegiado o protegido, rings), de la paginación de la RAM, de la codificación de ficheros…

3.1.4 Estructura de un SO

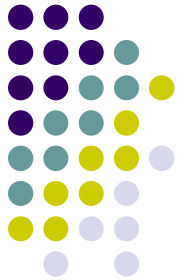


Los sistemas operativos, según su estructura, se suelen dividir en:

- Sistemas operativos monolíticos
- Microkernels
- Sistemas operativos estructurados (Win NT)

También mencionaremos, por el uso que haremos de ellas, la estructura de las máquinas virtuales

3.1.4 Sistema Operativo Monolítico



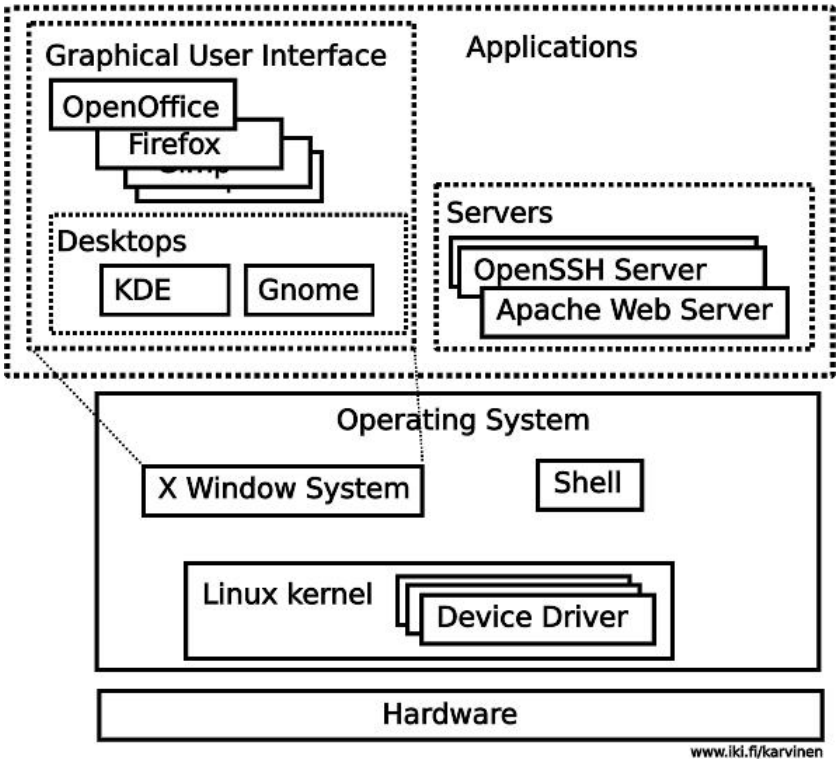
Características:

- No tienen estructura definida
- Todos los componentes (gestión de I/O, de usuarios, procesos, memoria) se realizan por el mismo programa (el SO)
- Todas las funciones se ejecutan en modo privilegiado

3.1.4 Sistema operativo monolítico

Ejemplos:

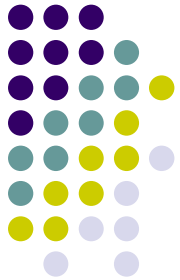
- MSDOS
- Unix, Linux (el kernel hace de pieza que controla todas las funciones del SO)



Empezaron como pequeños sistemas de uso particular, que al popularizarse fueron creciendo de forma no estructurada

Estructura del SO Linux (centrada en el kernel)

3.1.4 Sistema operativo monolítico

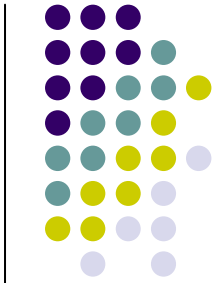


Desventajas

- Difícil añadir o modificar funciones
- Compuestos por miles o millones de líneas en un (o en pocos) programa
- No hay ocultación de la información (siempre modo privilegiado)

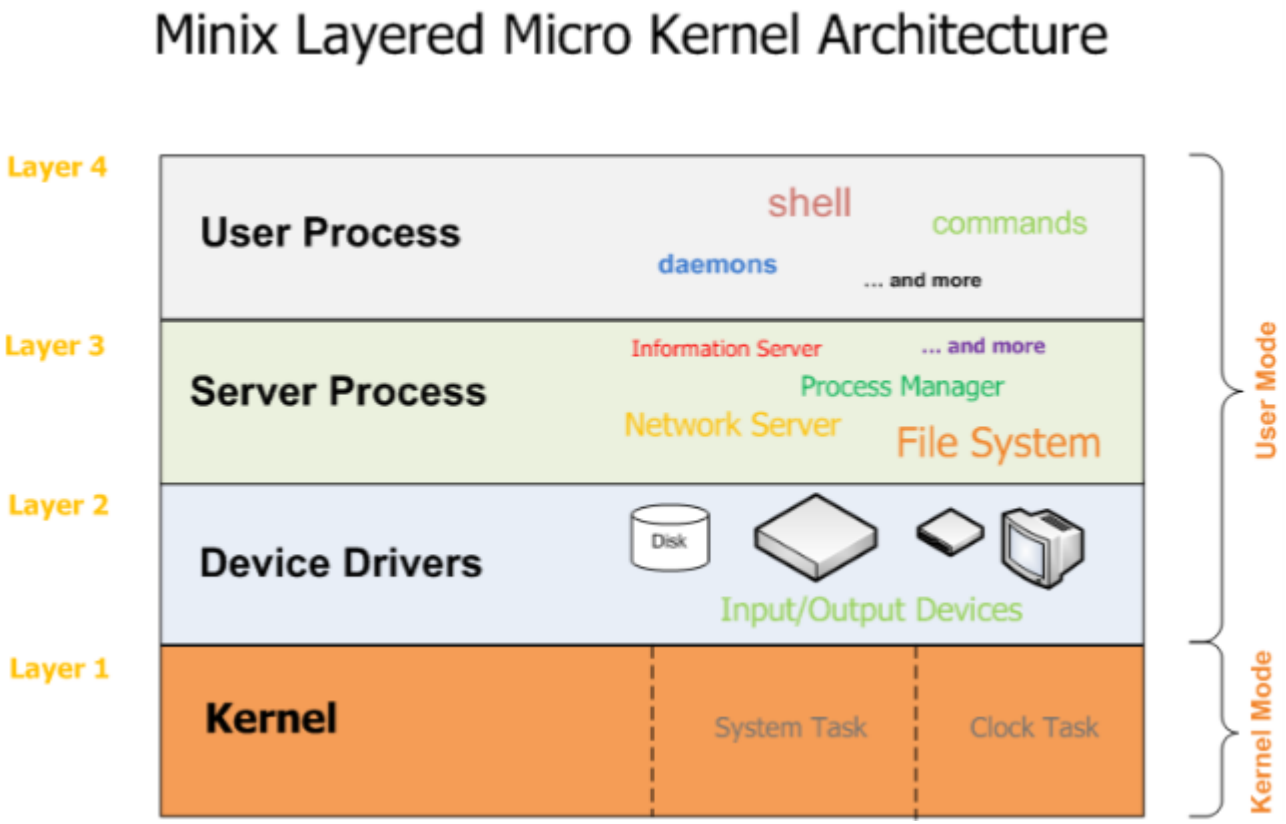
Para hacerlos mas útiles, y más fáciles de mantener y desarrollar, debemos dotarlos de algo de estructura

3.1.4 Microkernels



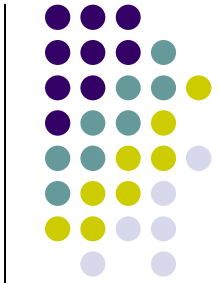
- Ponen el énfasis en la fiabilidad (para dispositivos donde la seguridad es la máxima prioridad)
- Minix (creado por Andrew S. Tanenbaum) o Symbian son buenos ejemplos
- Las distintas tareas del SO se implementan por modo de “servidores” de gestión de ficheros, drivers, procesos, red…

3.1.4 Microkernels



Estructura del SO Minix, con un kernel mínimo sobre el que funcionan, en “modo usuario”, la mayor parte de las funciones del SO

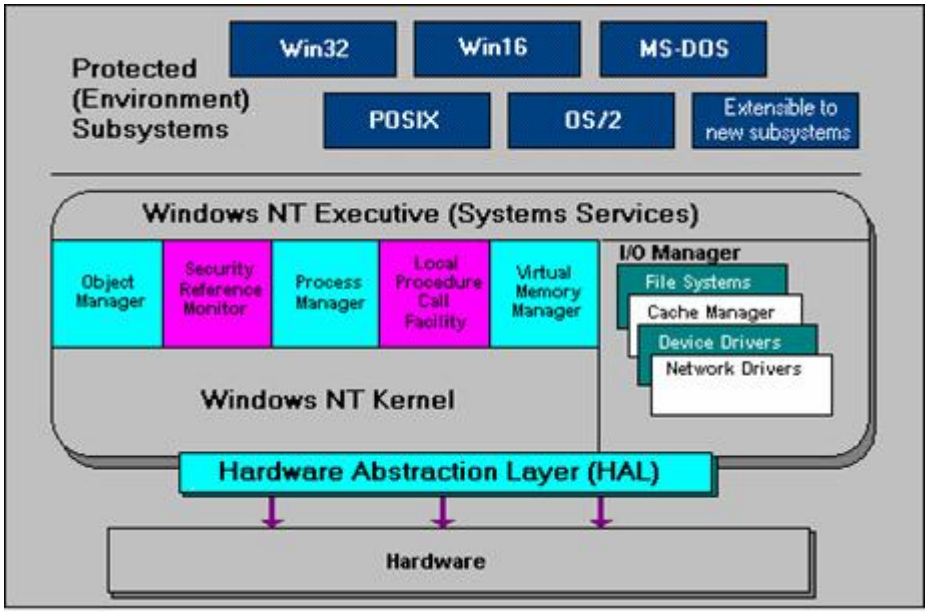
3.1.4 Sistema operativo estructurado (Win NT)



Sólo nos detendremos en la arquitectura cliente/servidor.
Características:

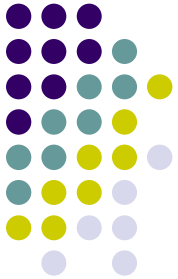
- La mayor parte de operaciones se ejecuta como procesos de modo protegido (llamados servicios)
- Sólo algunas acciones se ejecutan en modo privilegiado (micronúcleo)
- El micronúcleo suele encargarse de gestión de interrupciones, procesos, memoria y comunicación básica entre procesos
- No es un “microkernel” puro, ya que integra las funciones de “microkernel” (gestión de procesador y arquitectura del sistema) y el modo “executive” (que gestiona entrada/salida, procesos…)

3.1.4 Sistema operativo estructurado (Win NT)



Estructura de Windows NT (observar los Servicios de Sistema, “System Services”, propios de la arquitectura cliente/servidor)

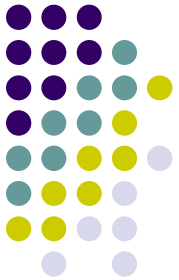
3.1.4 Sistema operativo estructurado



Funcionamiento

- Las peticiones del usuario se entienden como clientes de los servicios que presta el SO. Cada servicio puede demandar servicios adicionales
- Los servicios modularizan la estructura del SO y permiten que los fallos en un servicio sólo afecten a ese módulo

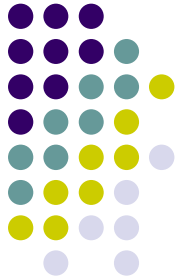
3.1.4 Sistema operativo estructurado



Inconvenientes:

- Mayor sobrecarga en el tratamiento de los servicios, ya que cada servicio se ejecuta en un espacio distinto

3.1.4 Máquinas virtuales



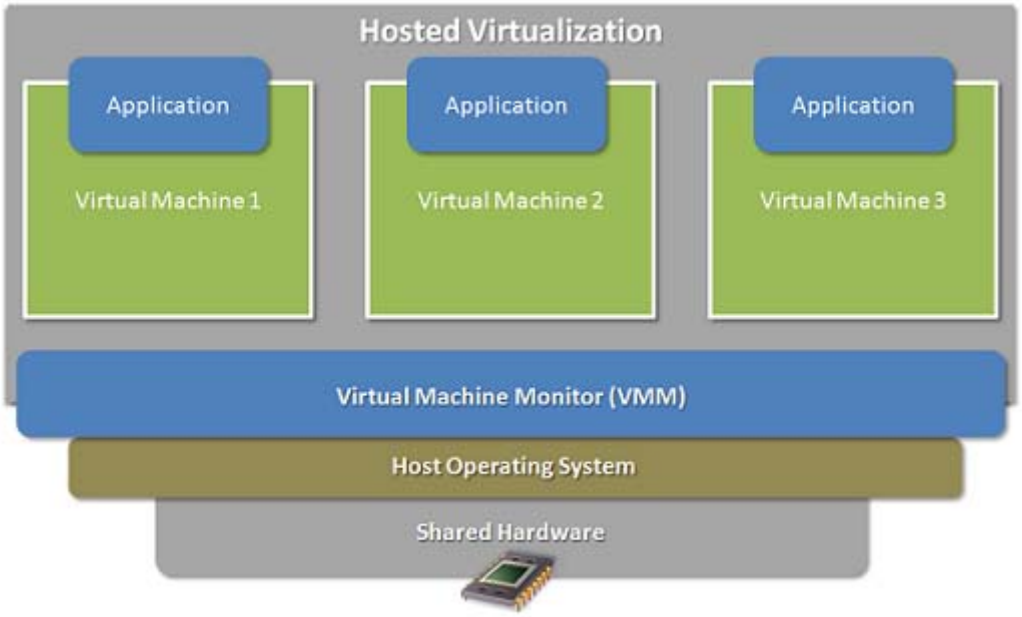
Consisten en una implementación por software de una máquina que ejecuta instrucciones como si fuera una máquina física. Pueden ser de dos tipos:

- Virtualizaciones de hardware
- Virtualizaciones de aplicaciones

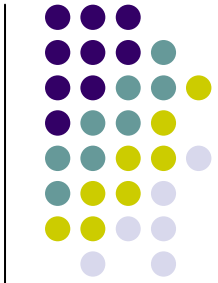
3.1.4 Virtualización de hardware

- Permiten compartir los recursos físicos de la máquina subyacente entre diferentes máquinas virtuales (cada una con su SO)
- "Esconde" el hardware original, permitiendo usar las máquinas virtuales creadas
- Cada máquina virtual se ejecuta sobre un "monitor"

Ejemplos:
VMware, Virtual PC, VirtualBox

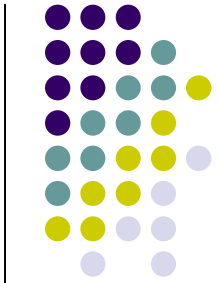


3.1.4 Virtualización de aplicaciones



- Una máquina virtual de aplicación permite “esconder” las características propias del SO, haciendo que dicha aplicación se ejecute directamente sobre la máquina virtual
- Permite la portabilidad de la aplicación a distintos SO
- Ejemplos: JVM (Java Virtual Machine), Flash Player

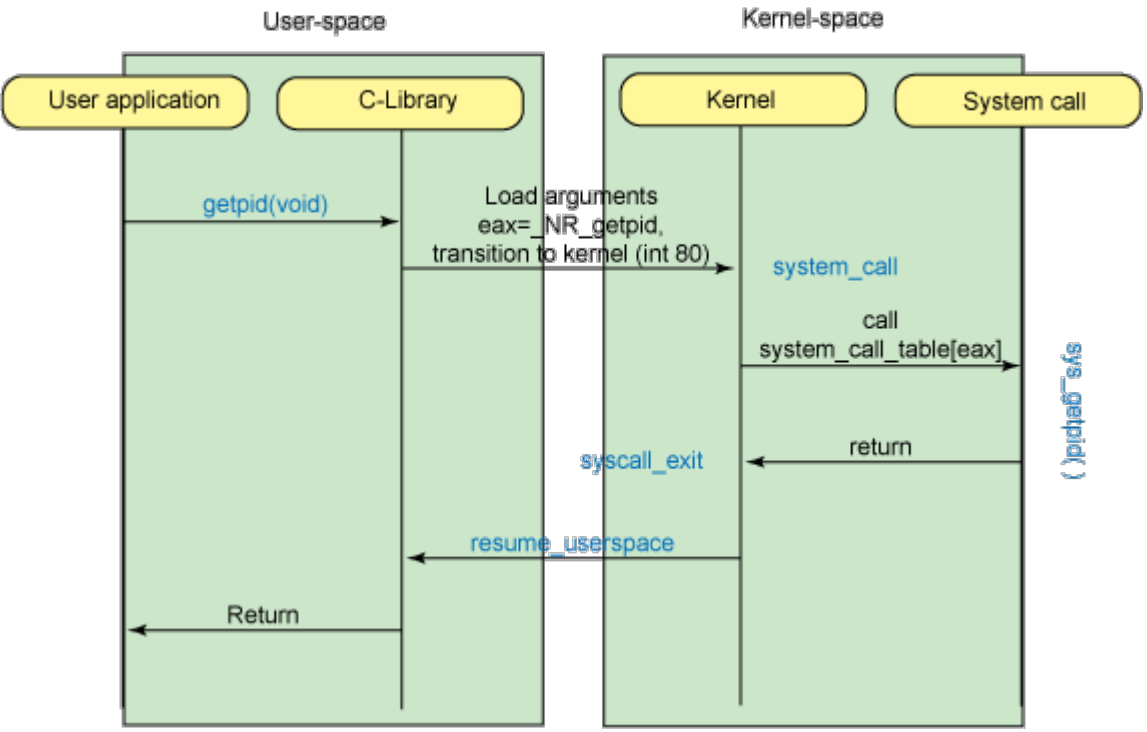
3.1.5 Llamadas al sistema



- Las llamadas al sistema (syscall) son la forma en que los programas requieren servicios del kernel del SO (recordar el modo usuario y el modo privilegiado)

- Entre los servicios solicitados al kernel (serán dependientes del SO) nos podemos encontrar 5 categorías:
  - Control de procesos
  - Gestión de ficheros
  - Gestión de dispositivos
  - Información de mantenimiento
  - Comunicación

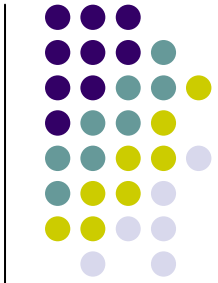
3.1.5 Llamadas al sistema (en Linux)



Flujo simplificado de una llamada de sistema con interrupciones;

- La función invocada es “getpid()”;
- La librería C busca la llamada de sistema correspondiente y los argumentos
- Se genera una interrupción, que produce la system_call
- Después el proceso inverso devuelve el valor resultante

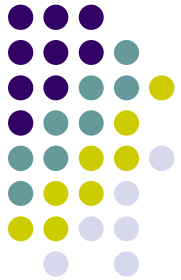
3.1.5 Llamadas al sistema (en Linux)



Dependen de la versión del kernel, una posible lista está en http://asm.sourceforge.net/syscall.html:

- Control de procesos:
  - Cargar, ejecutar (sys_execve), crear, terminar (sys_exit), observar o modificar los atributos (sys_getpid), alojar o liberar memoria
- Gestión de ficheros:
  - Crear (sys_creat), eliminar, abrir (sys_open), cerrar (sys_close), leer (sys_read), editar (sys_write) o modificar atributos
- Gestión de dispositivos
  - Solicitar o liberar dispositivo (sys_creat), leer (sys_open), escribir, modificar atributos… (sys_mount, sys_umount)

3.1.5 Llamadas al sistema (en Linux)

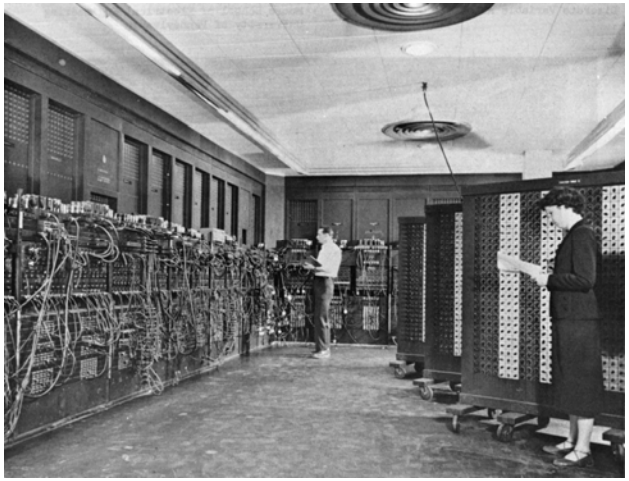


- Información de mantenimiento:
  - Obtener o modificar fecha/hora (sys_time, sys_stime), obtener/modificar datos de sistema, obtener/modificar atributos de procesos (sys_getpid), ficheros (sys_utime) o dispositivos
- Comunicación
  - Crear/borrar conexiones (sys_connect), enviar/recibir mensajes…

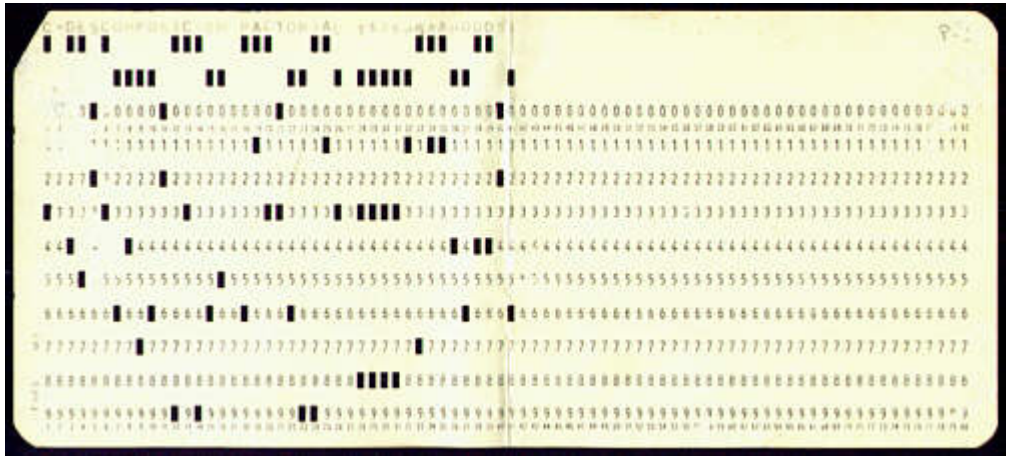
3.2 Algunos ejemplos de SO

Un poco de historia sobre los SO:

- 1940 – 1950: Primeros ordenadores (ENIAC, EDVAC). No disponen de SO. Los programas se codifican en instrucciones máquina. La entrada era por tarjetas perforadas, la salida por cinta de papel

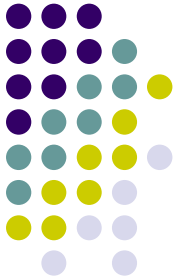


Computadora ENIAC (1946)



Tarjetas perforadas

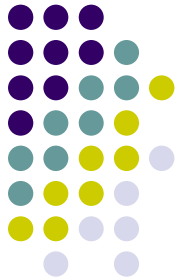
3.2 Un poco de historia sobre los SO:



1950 – 1960. Algunas características de los SO:

- Procesaban un único flujo de trabajo en lotes
- Disponían de rutinas de E/S
- Usaban mecanismos que facilitaban pasar de un trabajo a otro
- Permitían la recuperación del sistema si un trabajo no concluía satisfactoriamente

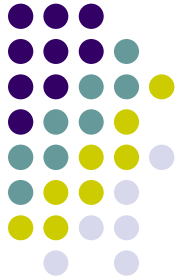
3.2 Un poco de historia sobre los SO:



1960 – 1970: empiezan a surgir algunas de las ideas presentes en los ordenadores actuales:

- Multiprocesadores
- Operaciones de E/S por acceso directo a memoria, o DMA (sin intervención de la CPU)
- Primeros sistemas de tiempo compartido (permitiendo trabajar a varios usuarios simultáneamente)
- Sistemas OS/360 de IBM, MULTICS de IBM y BELL Laboratories

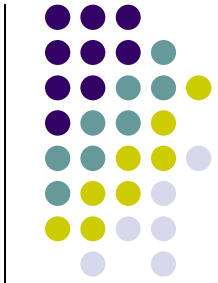
3.2 Un poco de historia sobre los SO:



1970 – 1980: aparición de los primeros SO de propósito general:

- Primera versión disponible de Unix (programado en C, sólo un pequeño núcleo en ensamblador)
- Las distintas versiones de Unix (como BSD, 1982) incluyen ideas como la memoria virtual o sockets para aplicaciones TCP/IP

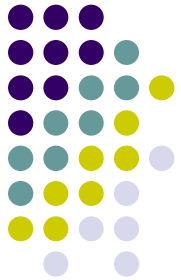
3.2 Un poco de historia sobre los SO:



1980 - …: Múltiples mejoras, incluyendo:

- Máquinas virtuales que simulan hardware
- Bases de datos sustituyendo ficheros
- Aparición de los gestores de ventanas
- GNU/Linux, MSDOS, Windows …, MAC

3.2 Unix y las distribuciones Linux

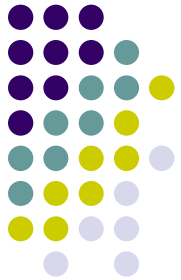


Unix nació en los años setenta con el propósito de implementar gestión de

- procesos,
- memoria,
- ficheros,
- dispositivos externos,

dentro del núcleo del SO (pero con la mínima funcionalidad para permitir múltiples políticas para cada módulo)

3.2 Unix y las distribuciones Linux



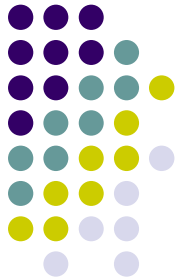
De este modo, generaron un kernel minimal y eficiente (en vez de un SO modular)

En la actualidad el sistema ha evolucionado a:

- permitir multiprocesadores y hardware distribuido
- soportar protocolos de red, dispositivos gráficos

El kernel Unix se ha vuelto complejo y bastante grande, pero sigue manteniendo la estructura monolítica

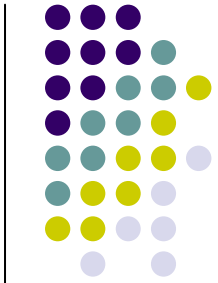
3.2 Unix y las distribuciones Linux



Desde su aparición en 1991, Linux se ha caracterizado por:

- ser una implementación robusta de Unix
- ser libre (aunque haya distribuciones de pago), General Public License (GPL)
- ser una implementación significativa del conjunto de llamadas al SO POSIX
- estar desarrollado por un grupo de colaboradores que trabajan de forma desinteresada

3.2 Unix y las distribuciones Linux

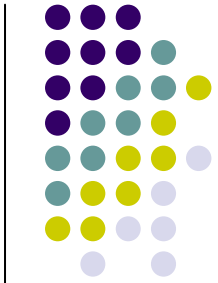


Linux se caracteriza también por disponer de un núcleo monolítico

GNU nace (años 80) del propósito de desarrollar programas libres. Dentro de esta ambición, se desarrollaron editores de texto (emacs, vi), terminales (bash), compiladores (g++, gcc), visores de imágenes (gimp)…

De la conjunción del kernel Linux y de las aplicaciones GNU nació el SO GNU/Linux

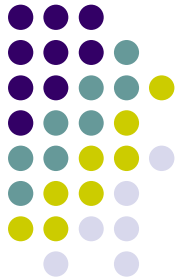
3.2 Unix y las distribuciones Linux



GNU/Linux se puede conseguir a través de programas libres:

- El kernel (http://www.kernel.org/)
- Las aplicaciones GNU (http://www.gnu.org/software/software.es.html)
- El sistema de ventanas (http://www.xfree86.org/)
- El entorno gráfico KDE (http://www.kde.org/)
- El entorno gráfico GNOME (http://www.gnome.org/)
- …

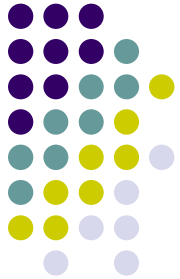
3.2 Unix y las distribuciones Linux



O también a partir de distribuciones con todos los anteriores programas ya disponibles (y probados juntos):

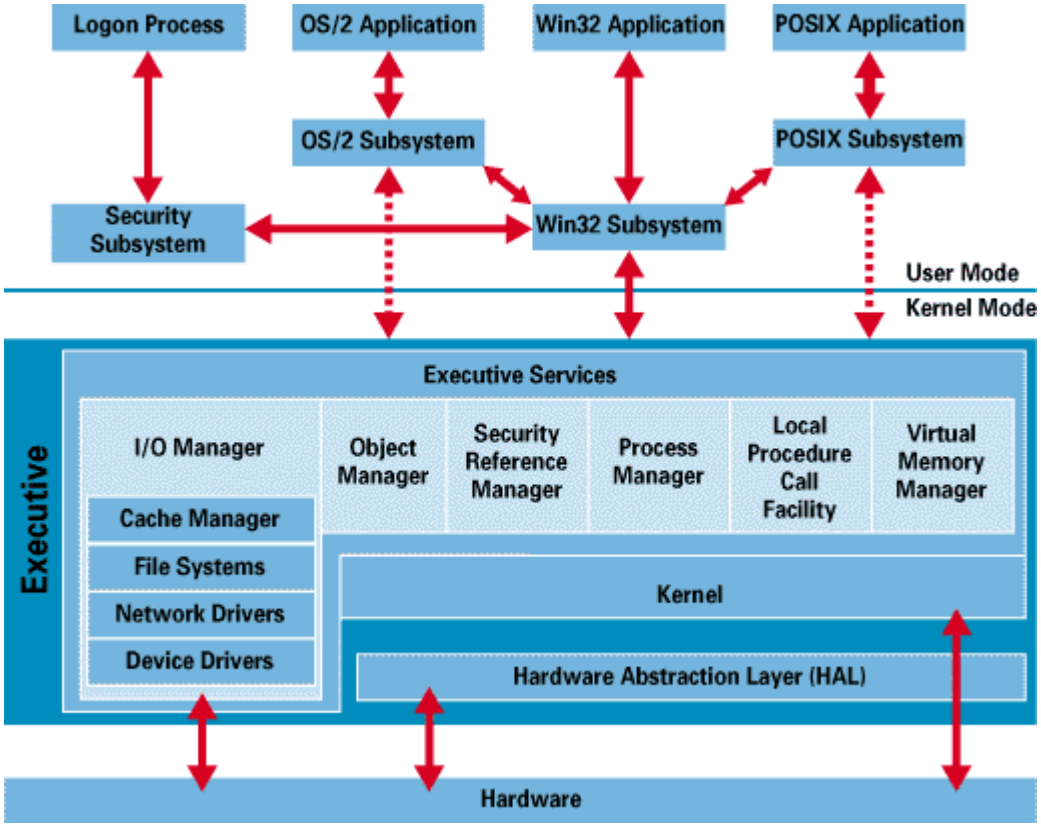
- Slackware (http://www.slackware.com), Debian (http://www.debian.org/) sólo admiten software o programas libres
- Redhat (http://www.redhat.com/), SUSE (http://www.novell.com/linux/), Mandriva (http://www2.mandriva.com/es/), Ubuntu (http://www.ubuntu.com/) admiten programas y complementos propietarios

3.2 Windows



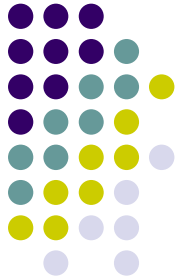
- Su primera versión (Windows 1.0) se remonta al año 1985
- En el año 1995 con Windows 95 se rediseña la interfaz gráfica
- Windows NT 3.1 (1993) trata de aportar soluciones a negocios (mayor seguridad, multiusuario…)
- A partir de Windows XP se fusionan la interfaz de Windows XP y las capacidades de Windows NT

3.2 Windows



La estructura del sistema se basa en capas (HAL, Kernel, Executive Services) que aportan distintas funcionalidades, y capas de abstracción sobre la parte del Usuario

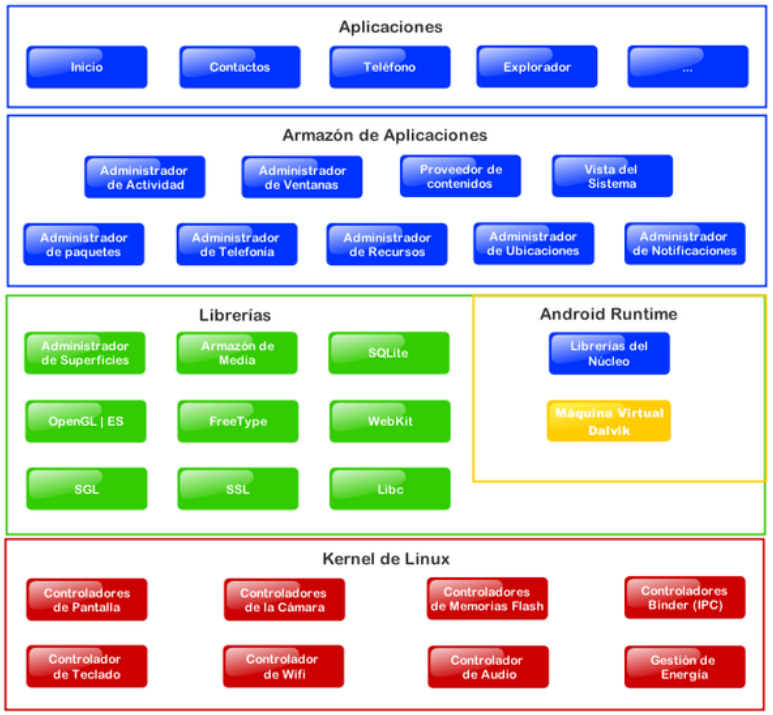
3.2 Mac OS



- Creado por Apple para ordenadores Macintosh
- Primer SO con una interfaz gráfica (1984)
- A partir de la versión Mac OS X, el SO se deriva de Unix pero manteniendo la interfaz gráfica

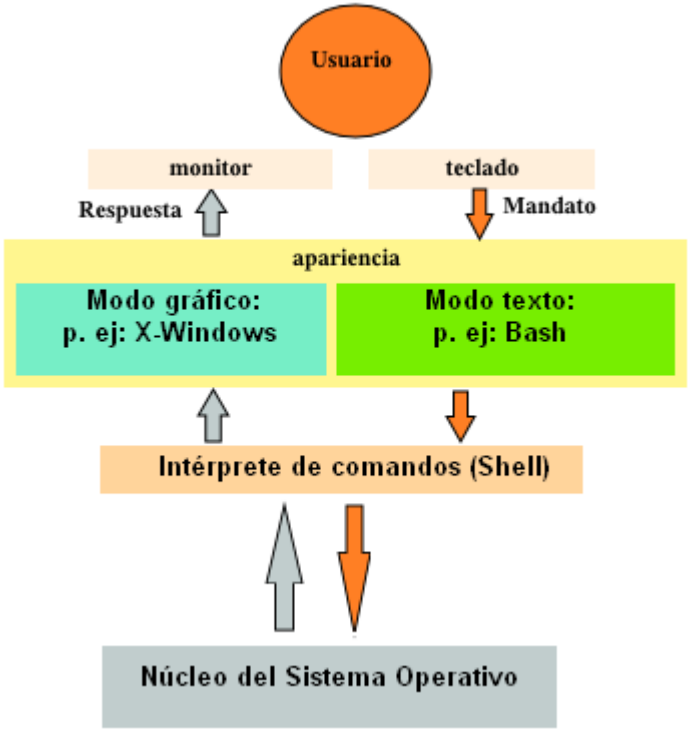
3.2 Android (SO móviles)

- Basado en una versión modificada del kernel Linux
- Cada proceso corre una instancia de la máquina virtual Dalvik
- Cada proceso es una aplicación Java



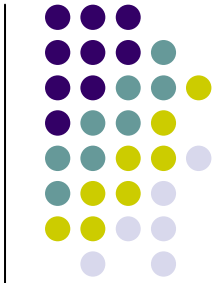
3.2 Tipos de interfaces habituales en un SO: “shell”, entorno de ventanas

Ejemplo de interacción usuario-máquina en un sistema operativo GNU/Linux



En general, para cualquier SO, siempre habrá una interfaz entre usuario y núcleo (de tipo Command Line Interface, CLI, si es de línea de mandatos, o de tipo GUI, interfaz gráfica de usuario)

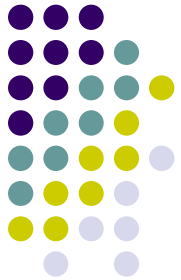
3.2 Shell o CLI



En general una shell (en Linux) nos permitirá realizar muchas de las tareas propias del SO:

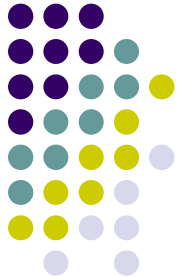
- Definir variables de entorno
- Definir alias para las órdenes más habituales
- Tiene algunos caracteres especiales, o metacaracteres, que interpreta de manera particular para realizar funciones específicas
- Algunas combinaciones de teclas las interpreta de un modo específico
- Tiene un lenguaje de programación propio
- Interpreta lo que escribimos según la sintaxis establecida y, en caso de que pretendamos ejecutar un comando o programa, lo busca y se lo entrega al núcleo para que lo ponga en ejecución (el shell es el padre de todos los procesos que ejecutes dentro de su entorno)
- Dispone de comandos internos (por ejemplo cd, echo, set, ...)
- Tiene caracteres y palabras reservadas para usos propios

3.2 Shell o CLI



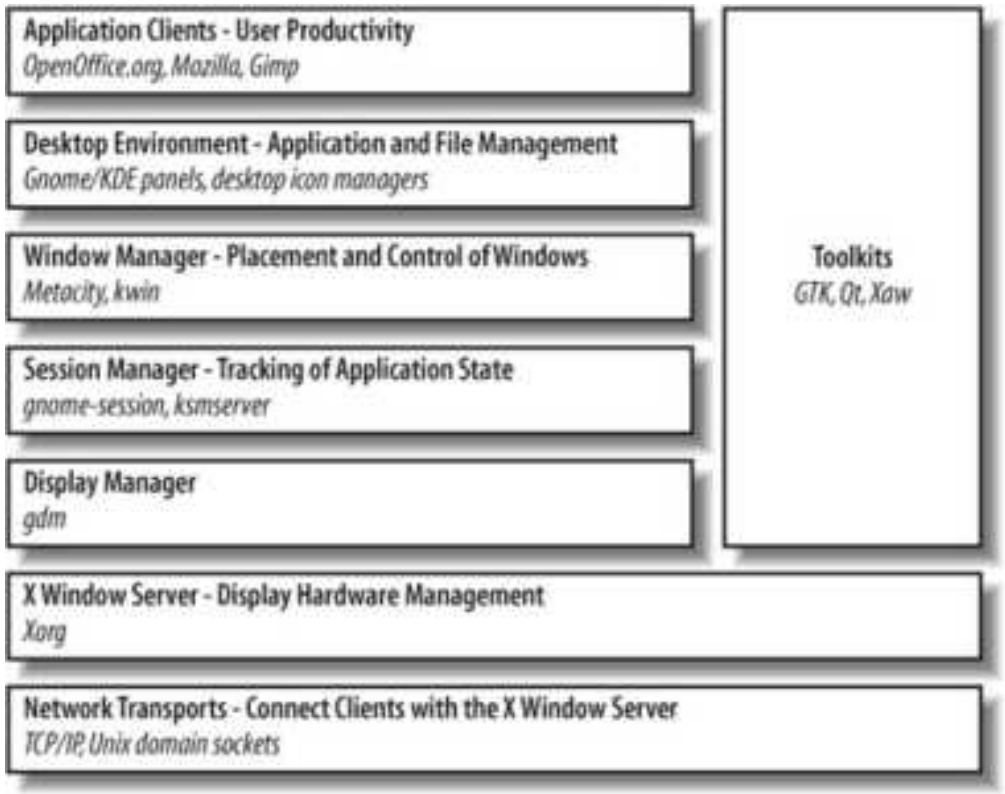
- La shell más común de Linux es la conocida como “bash” (es una renovación de “shell”, que añade autocompleción de mandatos…)
- Otras implementaciones comunes de la shell son “tcsh”, tc shell, versión extendida de la “c shell” con compleción de mandatos, editor de los mismos…, Korn shell “ksh”, típica de sistemas Linux y con mejor rendimiento que “bash”)…
- La diferencia entre unas y otras reside principalmente en el lenguaje o el conjunto de mandatos que soportan
- En Windows disponemos de “cmd.exe”

3.2 GUI o terminal gráfica



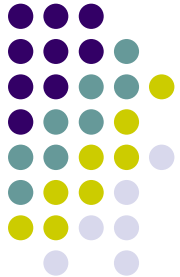
- En Windows es la opción por defecto
- En GNU/Linux hay varias capas de software responsables de la interfaz gráfica:
  - El sistema de ventanas X (habitualmente X11), independiente del SO. Se encarga de operaciones gráficas básicas como dibujar objetos, determinar la resolución de pantalla, profundidad de color, etc...
  - El gestor de ventanas, complementa las funcionalidades del sistema de ventanas X porque gestiona los bordes y botones y permite su movimiento , cierre, etc... Hay muchos gestores de ventanas disponibles para X11 entre los que podemos citar kwin (del entorno de escritorio KDE), Metacity (habitualmente usado por GNOME), etc...
  - El entorno de escritorio, es la capa más alta. Da un paso más y añade un gestor gráfico de ficheros para poder arrastrar y soltar, un panel para lanzar aplicaciones y muchas aplicaciones y utilidades propias

3.2 GUI o terminal gráfica



Niveles de la interfaz gráfica de ususario de Fedora. Solo “X Window Server” accede al hardware

3.2 CLI y GUI

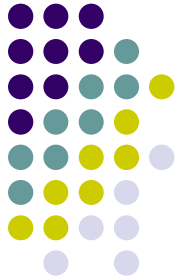


Algunos argumentos de la discusión “Command Line Interface” vs “Graphical User Interface”:

- La GUI hace que el SO sea más “amigable” con el usuario
  La GUI hace que el SO sea “vulgarizado”
- La CLI concede al usuario más control y funciones
  La CLI está anticuada
- Xwindows es sinónimo de evolución
  Xwindows presenta un gran riesgo de seguridad
- Usar la GUI es más rápido. Elegir y ejecutar iconos es más rápido que tratar de ejecutar mandatos
  Usar CLI es mucho más rápido. Por medio del teclado podemos trabajar mejor que en una GUI
- GUI consume demasiada CPU y memoria
  Con el desarrollo actual de los ordenadores, la diferencia no es significativa

Posiblemente, la ventaja está en usar ambas de una forma equilibrada y adecuada

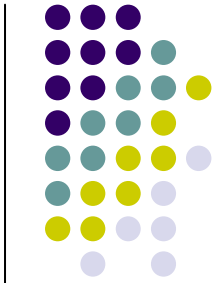
3.2 Gestión de entrada/salida, dispositivos externos, controladores



Una de las tareas fundamentales del SO es la gestión de la comunicación con los dispositivos externos (monitor, teclado, disco duro). Los dispositivos de E/S se pueden agrupar en tres categorías:

- Periféricos: permiten comunicar al usuario con el computador, tanto para entrada (ratón, teclado…) como para salida (impresoras, pantalla)
- Dispositivos de almacenamiento: proporcionan almacenamiento no volátil de datos (discos, sistemas de ficheros…)
- Dispositivos de comunicaciones: permiten conectar el ordenador a una red (modem, router, tarjeta de red…)

3.3 El sistema de archivos



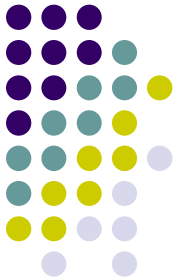
- Una de las tareas relevantes del SO es la gestión del sistema de archivos
- El sistema de archivos es el encargado de realizar las operaciones de gestión (apertura, cierre, modificación, copia) de ficheros en el sistema, de las operaciones que se pueden realizar sobre esos ficheros, del árbol de directorios, de permisos sobre los mismos (Sección 3.4), copias de seguridad…
- Entre las tareas que nos evita el sistema de ficheros está la de tener que acceder de forma directa a los dispositivos de almacenamiento, u ocuparnos de bloquear los ficheros en uso

3.3 El sistema de archivos

- Qué es: Un sistema de ficheros es una forma de organizar la información que debe ser almacenada cuando un programa termina (a diferencia de la memoria principal, que es volátil)
- Dónde se usa: Los sistemas de ficheros se usan en dispositivos de almacenamiento tales como discos duros, disquetes, CD's, memorias flash…

Volumen	Disposición	Tipo	Sistema de archivos	Estado	Capacidad	E
(C:)	Simple	Básico	NTFS	Correcto (Sistema, Arranque, Archivo de paginación, Activo, Volcado, Partición primaria)	169,25 GB	7
HP_RECOVERY (D:)	Simple	Básico	NTFS	Correcto (Partición primaria)	15,00 GB	5
HP_TOOLS (E:)	Simple	Básico	FAT32	Correcto (Partición primaria)	1,99 GB	1
FLASH DISK (G:)	Simple	Básico	FAT	Correcto (Activo, Partición primaria)	250 MB	1
Audio CD (F:)	Simple	Básico	CDFS	Correcto (Partición primaria)	0 MB	0

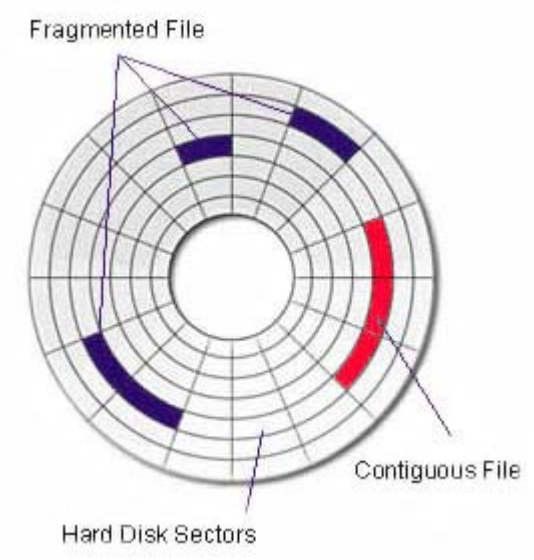
3.3 El sistema de archivos



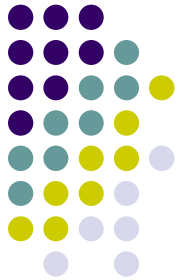
- Responsabilidades de un sistema de archivos:
  - Gestionar el espacio
  - Designar nombres de ficheros
  - Gestión de directorios o carpetas
  - Metainformación de ficheros
  - Diversas utilidades (desfragmentación…)
  - Seguridad
  - Mantener la integridad del sistema

3.3 El sistema de archivos

- Gestión del espacio
  - El sistema de ficheros es el encargado de organizar ficheros y directorios y guardar traza de qué áreas (sectores) de un dispositivo pertenecen a un fichero, y cuáles no están siendo usadas.



3.3 El sistema de archivos

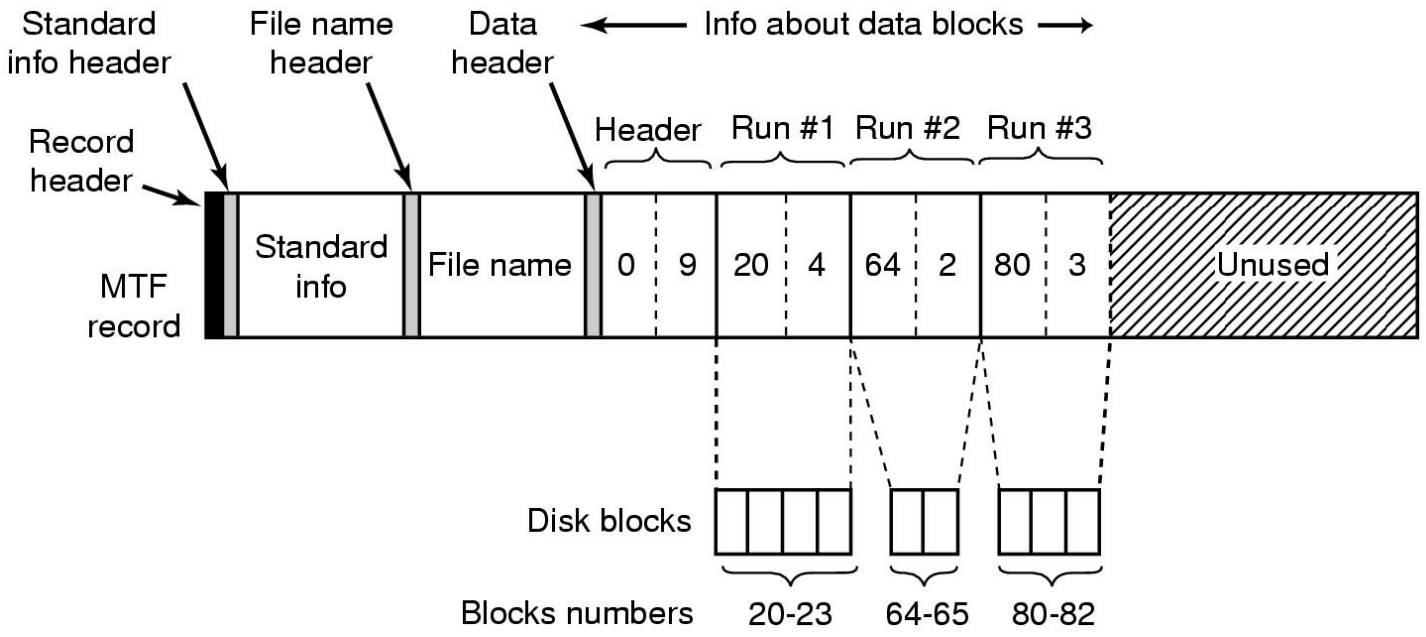


- Gestión del espacio

  Por ejemplo, en los sistemas de archivos NTFS (Win NT), la información sobre cada archivo se almacena en una tabla MFT, master file table (FAT, file allocation table, en los sistemas FAT y FAT32, visto en el tema 1)

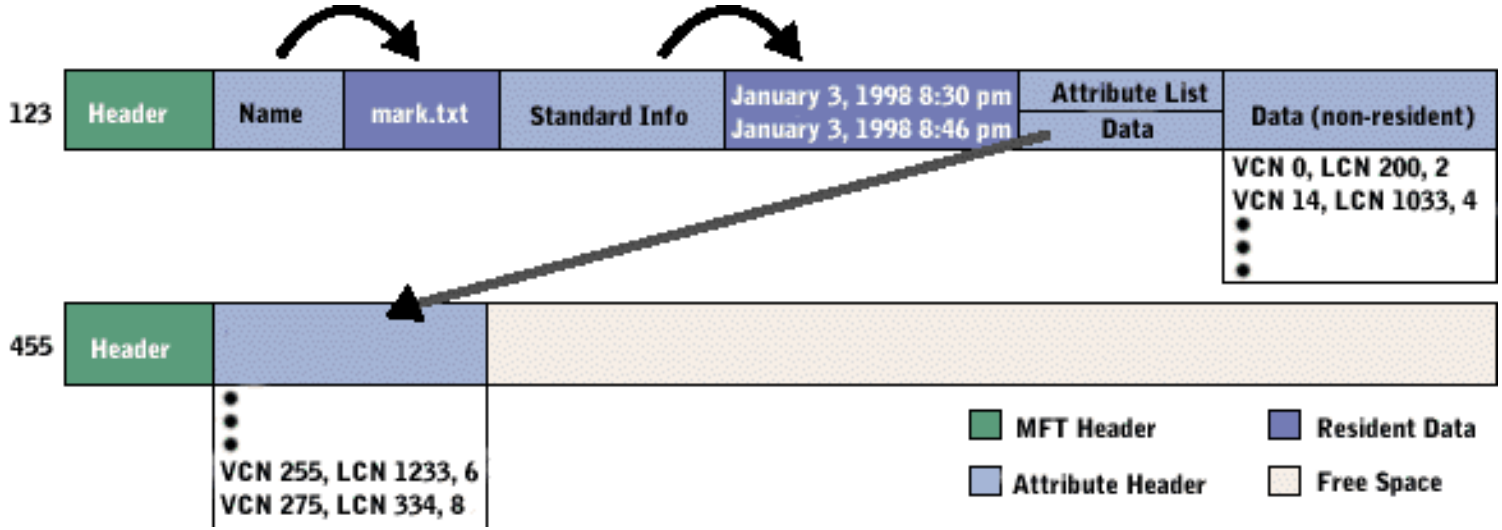
3.3 El sistema de archivos

La información sobre cada fichero se organiza como uno (o varios) registros (similares a los Inodos en ext3, ext4) en la tabla MFT

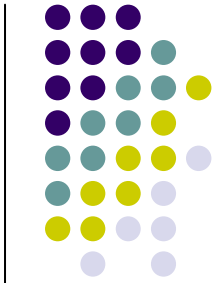


3.3 El sistema de archivos

- Ejemplo: un fichero “mark.txt”, con varios registros de atributos (123, 455), y cuyos datos empiezan en el LCN (logical cluster number) 200, con un “run” igual a 2 clústeres, sigue en el lcn 1033…



3.3 El sistema de archivos

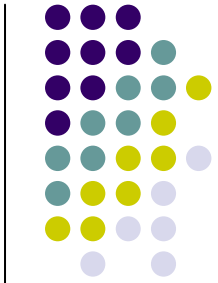


- Gestión del espacio
  - Eliminación de ficheros del disco duro

    Cuando “mandamos un fichero a la papelera” en realidad no estamos “limpiando” los clústeres que el mismo ocupa, simplemente los marcamos como “no usados”, y el SO los da por “potencialmente usables”; se podría recuperar accediendo directamente a los clústeres (“pasando” por encima del SO)

    Algunos programas o utilidades (gratuitos) permiten recuperar parte de esos ficheros, por ejemplo TestDisk (http://www.cgsecurity.org/wiki/TestDisk)

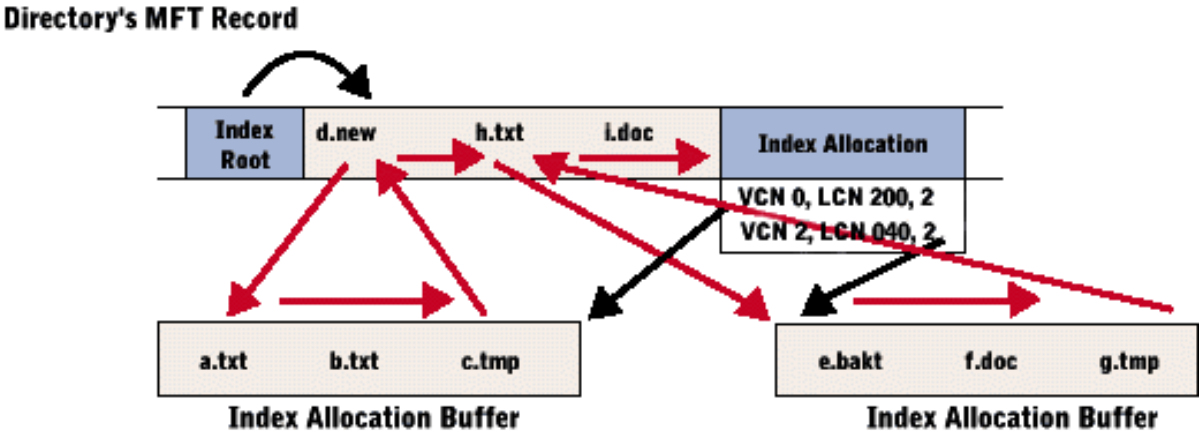
3.3 El sistema de archivos



- Designar nombres de fichero
  - El sistema de ficheros puede imponer restricciones sobre los nombres de ficheros que se pueden usar en el mismo tales como:
  - Longitud máxima de los nombres (en FAT16, 8 caracteres de nombre de fichero más 3 de la extensión, en FAT32 o NTFS hasta 255 caracteres…)
  - Caracteres empleados en los nombres (en NTFS se excluyen NULL y “/”, Win32 además elimina “\”, “:”, “*”, “?”,””””, “<” “>”…)
  - Segmentación especial de prefijos o sufijos en los nombres (extensiones en FAT: “hola.txt”, archivos ocultos en ext4: “.datos”), aunque algunas de éstas pueden provenir del SO, no del sistema de archivos

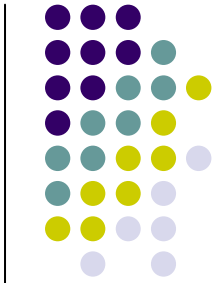
3.3 El sistema de archivos

- Gestión de directorios o carpetas
  - Generalmente, los sistemas de archivos permiten crear directorios o carpetas con el fin de agrupar archivos
  - Los directorios suelen ser, desde el punto de vista del sistema operativo, un “tipo especial” de fichero, que contiene la lista de ficheros, sus propiedades, y punteros a sus entradas en la “tabla de archivos”



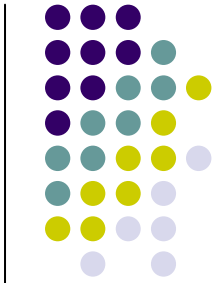
Directorio visto en la Master File Table de NTFS

3.3 El sistema de archivos



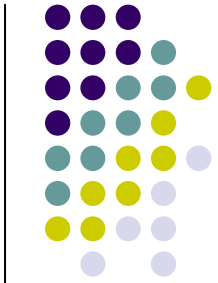
- Metainformación (o atributos) de ficheros
  - Tamaño del fichero (número de bloques o bytes, tamaño real, tamaño en el sistema de archivos)
  - Tiempo (o fecha) de creación, de último acceso, desde que los meta-datos fueron cambiados
  - Tipo de fichero (carpeta, fichero convencional, enlace simbólico, acceso directo, fichero de bloque, carácter…)
  - Propietario (userID) y grupo al que pertenece (groupID), lista de permisos, ACL (listas de control de acceso)

3.3 El sistema de archivos



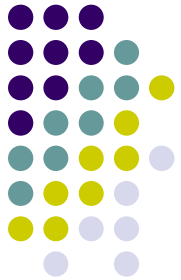
- Diversas utilidades
  - Dentro del sistema de ficheros, nos encontramos con “utilidades” que permiten crear (mkfile, touch), actualizar (touch), renombrar (mv, rename), eliminar (rm) los objetos que forman parte del mismo
  - Facilidades para desfragmentación, eliminar espacio en disco duro, gestionar las tablas de ficheros (FAT, MFT…)

3.3 El sistema de archivos



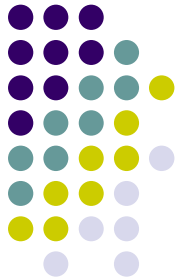
- Seguridad
  - A través de permisos, muchos de ellos basados en listas de control de acceso (ACL): ACL POSIX.1e en los sistemas Linux, ACL para NTFS en Windows, ACL NFSv4 para sistemas de archivos en red…
  - A través de contraseñas para ficheros
  - A través de encriptación de ficheros (posible en ext3 y en NTFS)

3.3 El sistema de archivos



- Mantener la integridad del sistema
  - Recuperar o mantener ficheros cuando existe algún fallo crítico del sistema o de programas modificando los ficheros o carpetas
  - Resolver o evitar operaciones anómalas (sobreescritura de ficheros)
  - Propiedades dinámicas del sistema de archivos: Bloquear ficheros cuando están en uso por otras aplicaciones…

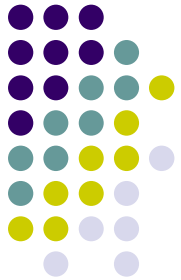
3.3 Ejemplos de sistemas de archivos



Algunos de los sistemas de archivos con los que hemos trabajado a lo largo del cuatrimestre:

- ext2, ext3, ext4
- NTFS
- FAT, FAT32 (vistos en el tema 1)

3.3 Sistemas de archivos ext3

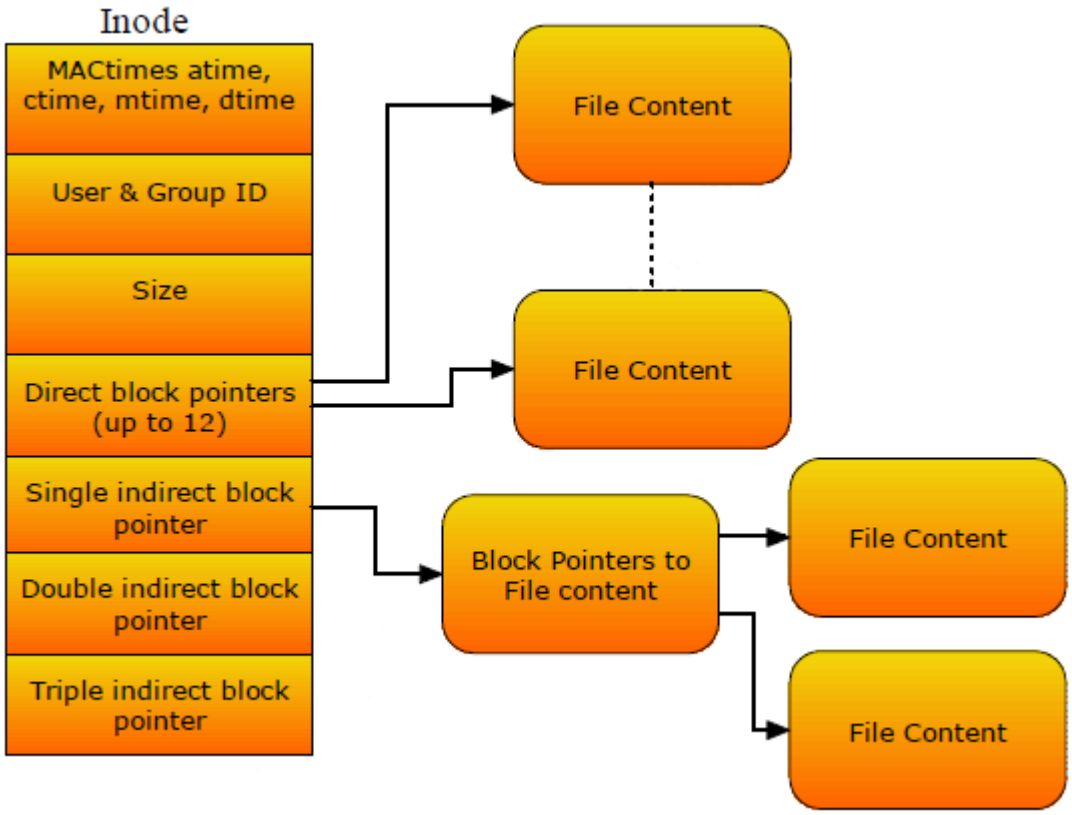


- El tipo de archivos más común en las distribuciones Linux es ext3 (y ext4). Sin embargo, el kernel Linux cuenta con VFS (Virtual Filesystem Switch), una interfaz que le permite gestionar de modo idéntico distintos sistemas de ficheros como ext2, ReiserFS, MSDOS, VFAT, NTFS, HFS, AFS… (y siguen aumentando)

- En realidad, es el VFS el que se encarga de gestionar los inodos y algunas de las llamadas básicas a sistema como “stat”, “chmod”, “chgrp”

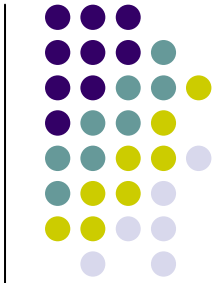
3.3 Sistema de archivos ext3

- Noción de inodo



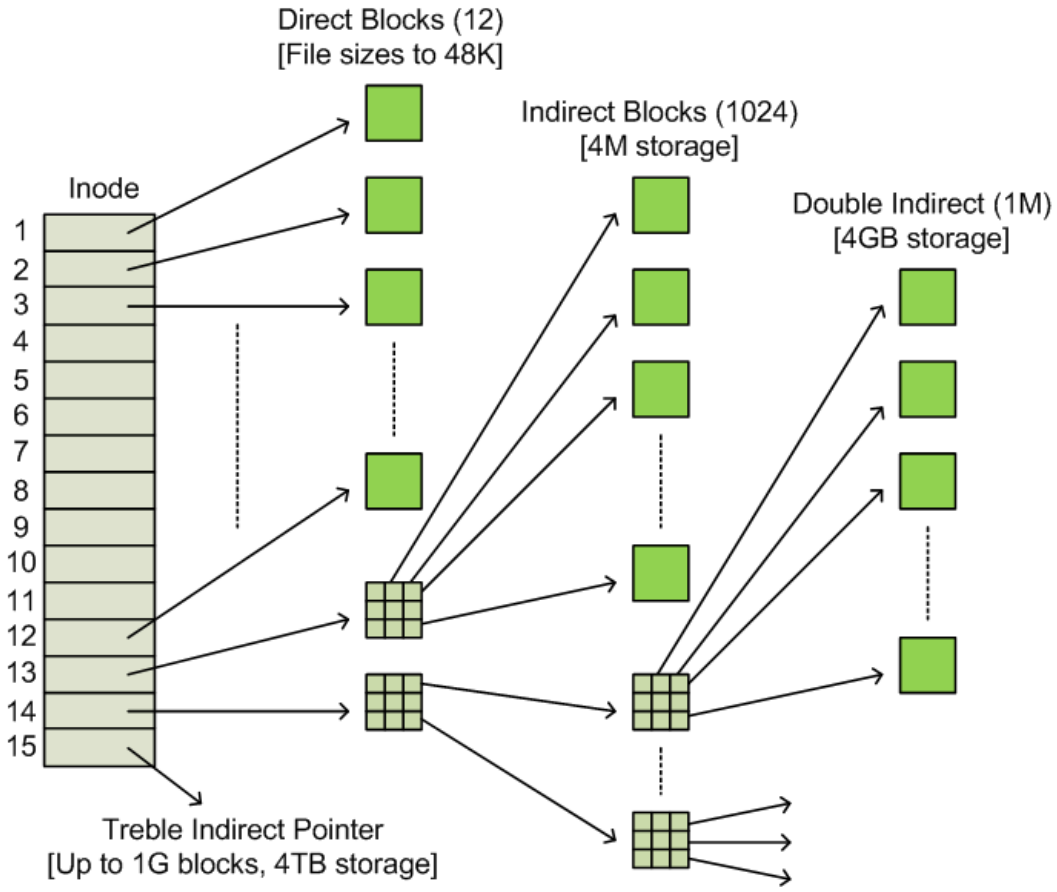
Por cada fichero o carpeta en el sistema, existe un inodo, una estructura de datos, que guarda la información del fichero. Es similar a los registros del MFT en NTFS

3.3 Sistema de archivos ext3



- Cada inodo ocupa 128 bytes en memoria; guarda información de:
  - Metainformación o atributos del fichero (fecha de acceso, modificación…)
  - Punteros (o direcciones de memoria a bloques) donde se aloja el contenido del fichero
    - Hasta 12 bloques (12*4KB = 48KB) se pueden guardar en el propio inodo
    - Si el fichero ocupa más de 12 bloques, el puntero apuntará a un bloque de punteros a los bloques del fichero (“indirección”)

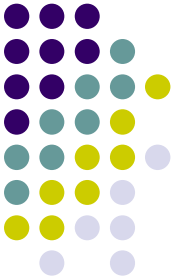
3.3 Sistema de archivos ext3



- Ficheros de menos de 48KB: se usa un único inodo para alojarlos
- Ficheros hasta 4MB: se usa un inodo que apunta a otro inodo, y éste a los bloques
- Ficheros hasta 4GB:…
- Ficheros hasta 4TB:…

(Indirecciones)

3.3 Sistema de archivos ext3



- Vista de un inodo desde el SO (por medio de stat):

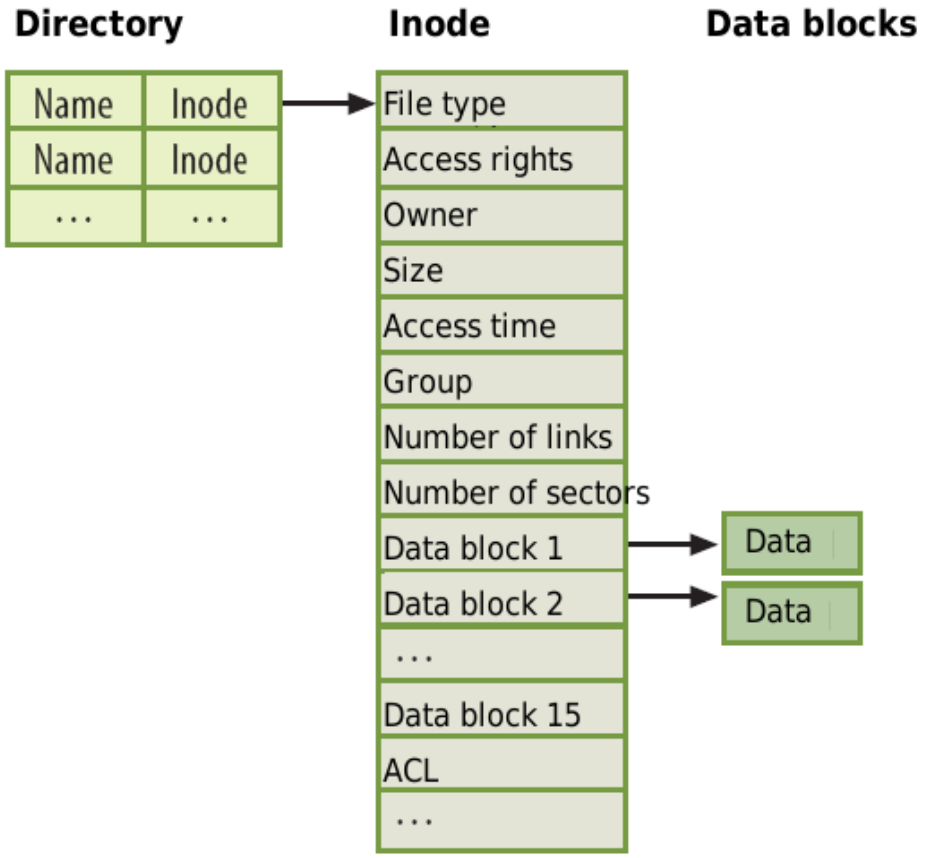
```
jesus@jesus-laptop:~/Escritorio$ stat enl_debil
  File: «enl_debil» -> «/usr/games/gnomine»
  Size: 18              Blocks: 0          IO Block: 4096   vínculo simbólico
Device: 801h/2049d      Inode: 305435      Links: 1
Access: (0777/lrwxrwxrwx)  Uid: ( 1000/   jesus)   Gid: ( 1000/   jesus)
Access: 2010-11-30 00:25:04.000000000 +0100
Modify: 2010-11-29 00:07:48.000000000 +0100
Change: 2010-11-29 00:07:48.000000000 +0100
```

- Se pueden observar los atributos (fechas, usuarios, permisos), el número del inodo, los bloques usados…

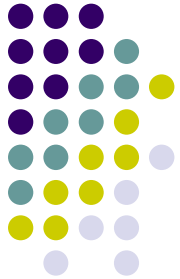
```
alumno@pc-alumno:/media/VBOXADDITIONS_4.1.0_73009$ stat VBoxWindowsAdditions-amd64.exe
  File: «VBoxWindowsAdditions-amd64.exe»
  Size: 13097968        Blocks: 25582      IO Block: 2048   fichero regular
Device: b00h/2816d      Inode: 1833        Links: 1
Access: (0555/-r-xr-xr-x)  Uid: ( 1001/  alumno)   Gid: ( 1001/  alumno)
Access: 2011-07-19 12:30:37.000000000 +0200
Modify: 2011-07-19 12:30:38.000000000 +0200
Change: 2011-07-19 12:30:38.000000000 +0200
alumno@pc-alumno:/media/VBOXADDITIONS_4.1.0_73009$
```

3.3 Sistema de archivos ext3

- Directorios:
  Desde el punto de vista de ext3, los directorios solo contienen nombres de ficheros, y los inodos de esos ficheros
- A la hora de ser alojados en disco duro, los directorios se alojan como ficheros cuyo contenido son otros ficheros

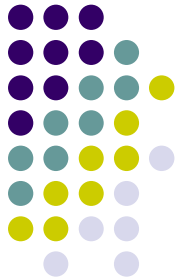


3.3 Sistema de archivos ext3



- Resumiendo, los tipos de estructuras de datos que encontramos en ext3 (ext2, ext4) son directorios, inodos y bloques

3.3 Sistema de archivos ext3



- Los 7 tipos de ficheros que podemos encontrar en Linux (no en ext3), son:
  - ficheros regulares (-): de texto, binarios, código fuente, comprimidos… Dan acceso a uno o varios bloques de memoria
  - directorios (d)
  - enlaces simbólicos (l): proveen accesos a otros ficheros, carpetas… Guardados en memoria por inodos, dan acceso a otros ficheros, no a los bloques

3.3 Sistema de archivos ext3

- ficheros de dispositivos de bloque (b): para representar dispositivos de bloque (particiones de discos duros, usbs…)

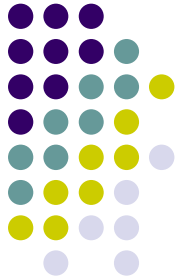
```
brw-rw----   1 root    disk      8,   0 2011-11-23 11:35 sda
brw-rw----   1 root    disk      8,   1 2011-11-23 11:35 sda1
brw-rw----   1 root    disk      8,   2 2011-11-23 11:35 sda2
brw-rw----   1 root    disk      8,   5 2011-11-23 11:35 sda5
```

- ficheros de dispositivos de carácter (c): para representar dispositivos de carácter (terminales, salidas de mandatos…)

```
crw-rw-rw-   1 root    tty       5,   0 2011-11-23 12:18 tty
crw--w----   1 root    root      4,   0 2011-11-23 11:35 tty0
crw-------   1 root    root      4,   1 2011-11-23 11:35 tty1
crw--w----   1 root    tty       4,  10 2011-11-23 11:35 tty10
crw--w----   1 root    tty       4,  11 2011-11-23 11:35 tty11
```

- sockets (s): intercomunicación de procesos
- pipes (p): intercomunicación de procesos

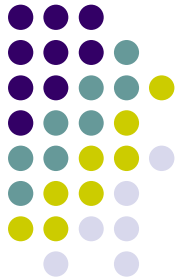
3.3 Sistemas de archivos ext3



Algunas limitaciones prácticas de ext3:

- Máxima longitud de nombre de fichero: 254 caracteres
- Máximo tamaño de unidad: de 2 a 16TB (dependiendo del tamaño de bloque)
- Máximo de 32000 subdirectorios por directorio
- Permisos: los propios de sistemas Linux (r, w, x, s, S)
- Permite conversión directa de las particiones ext2 a formato ext3
- Dispone de journaling, que permite guardar traza de los cambios realizados sobre un fichero y recuperar versiones originales bajo fallos del sistema

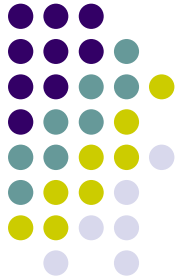
3.3 Sistemas de archivos ext3 (ext4)



Algunas características básicas sobre ext4:

- Tamaño máximo de archivo: 16TB
- Tamaño máximo de sistema: 10^6 TB
- Máximo número de ficheros: 4 billones
- Permisos: los propios de POSIX (o UNIX)
- Es compatible con ext3 y ext2
- Dispone de journaling

3.3 Sistemas de archivos ext3

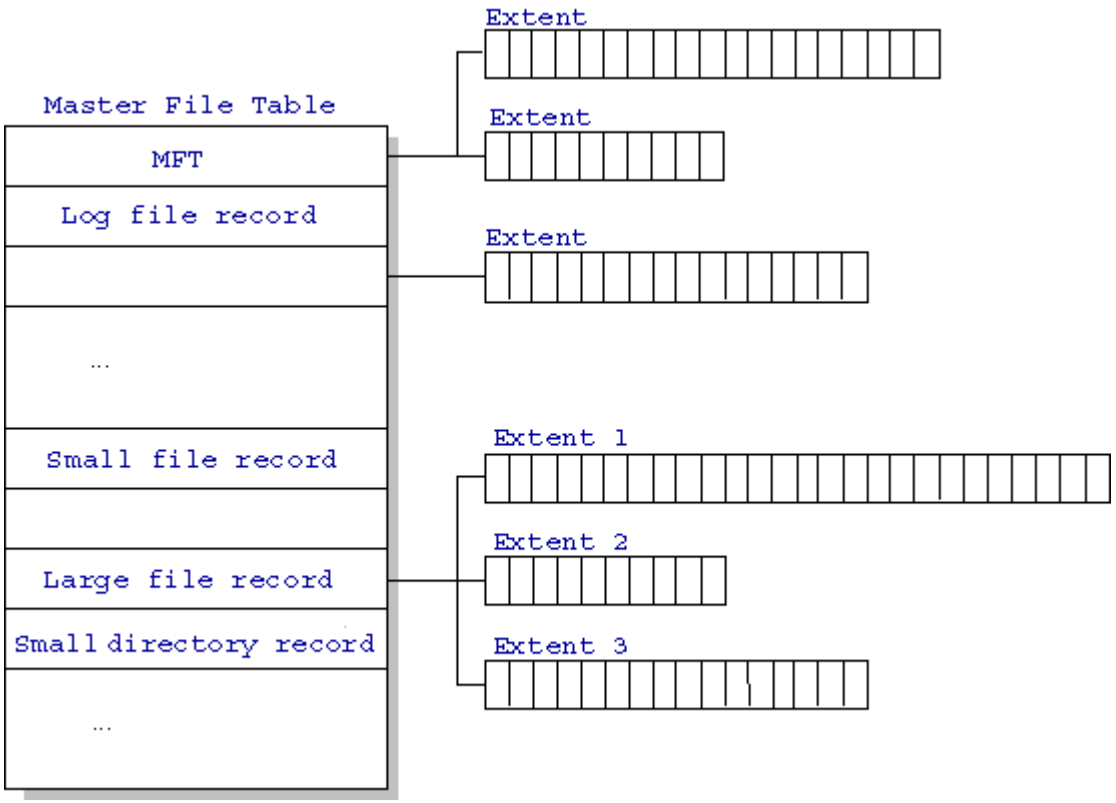


Algunas utilidades:

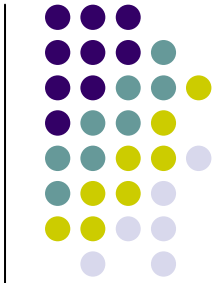
- stat: muestra el estatus de un fichero
- file nombre_de_fichero: permite conocer el contenido de un fichero (si el mismo es un binario, texto plano, texto con formato html, xml…)
- fdisk: gestionar tablas de particiones
- mount: montaje de dispositivos en el sistema de ficheros
- umount: desmontaje de unidades

3.3 Sistema de archivos ntfs

- Basado en el uso de la MFT (master file table)



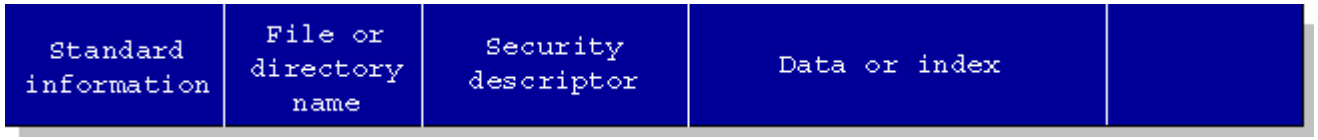
3.3 Sistema de archivos ntfs



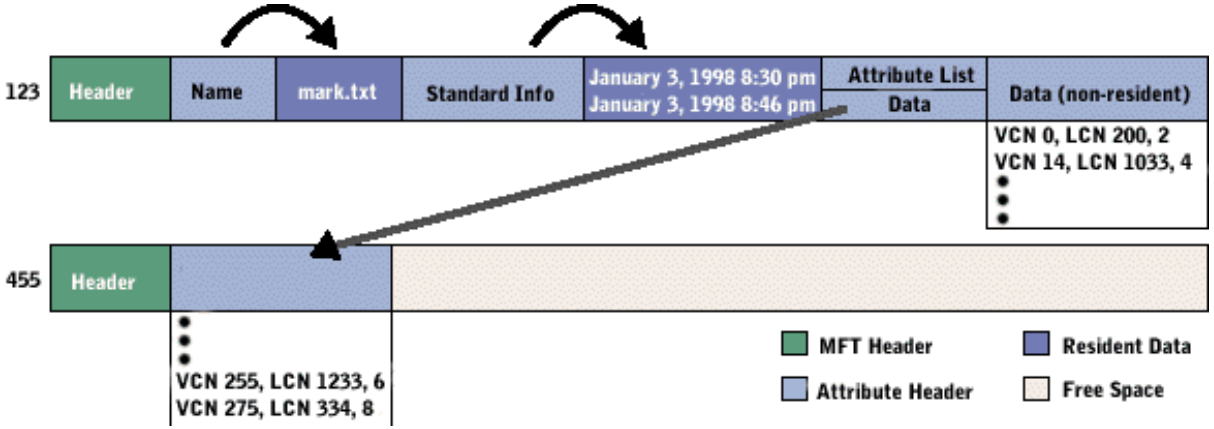
- Características
  - Cada fichero y directorio del sistema está representado por medio de un “record” (o registro) en la MFT
  - Los ficheros o directorios “pequeños” (de menos de 512 bytes), directamente están alojados en ese registro, optimizando su velocidad de búsqueda
  - Los ficheros “grandes” necesitan de unos registros “extendidos” (extent) donde aparecen todas las direcciones de memoria que el fichero ocupa
- Algunos programas gratuitos, como NTFSWalker (http://dmitrybrant.com/ntfswalker), permiten “explorar” las entradas de la tabla MFT para cada archivo

3.3 Sistema de archivos ntfs

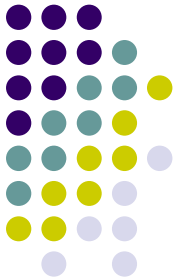
- ¿Qué aspecto tiene cada entrada (record) de la MFT?



En el caso de un fichero “largo”, el registro nos da la dirección de otros registros donde se alojan los contenidos del fichero



3.3 Sistema de archivos ntfs

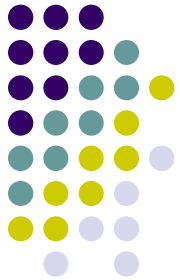


Aspecto de un “record” de MFT con algunos de los atributos más comunes

$MFT Entry Header	$STANDARD_INFO	$FILE_NAME	$DATA
(42 bytes in length)	(0-3) Type=16 (0x10)	(0-3) Type=48 (0x30)	(0-3) Type=128 (0x80)
(0-3) Signature: FILE (0x46494c45)/BADD (0x44414142) (16-17) Sequence Value (18-19) Link Count (number of names the entry has) (20-21) Offset to First Attribute (22-23) Flags: In-use (0x01), Directory (0x02) (32-39) Base Record Reference (if 0 then it is a base entry) (40-41) Next Attribute ID	(0-7) $SI Creation (born) Time (8-15) $SI Modification Time (16-23) $SI Entry Time (24-31) $SI Access Time (32-35) Flags: Read Only (0x0001), Hidden (0x0002), System (0x0004), Archive (0x0020), Compressed (0x0800), Encrypted (0x4000)	(0-7) Parent Directory (8-15) $FN Creation (born) Time (16-23) $FN Modification Time (24-31) $FN Entry Time (32-29) $FN Access Time (65) POSIX, WIN32, DOS, Win32 & DOS (0-3) (66+) File Name *Note: May have up to four entries of each MACE record	File Data (if under 700 Bytes) or Index to Non-resident attribute

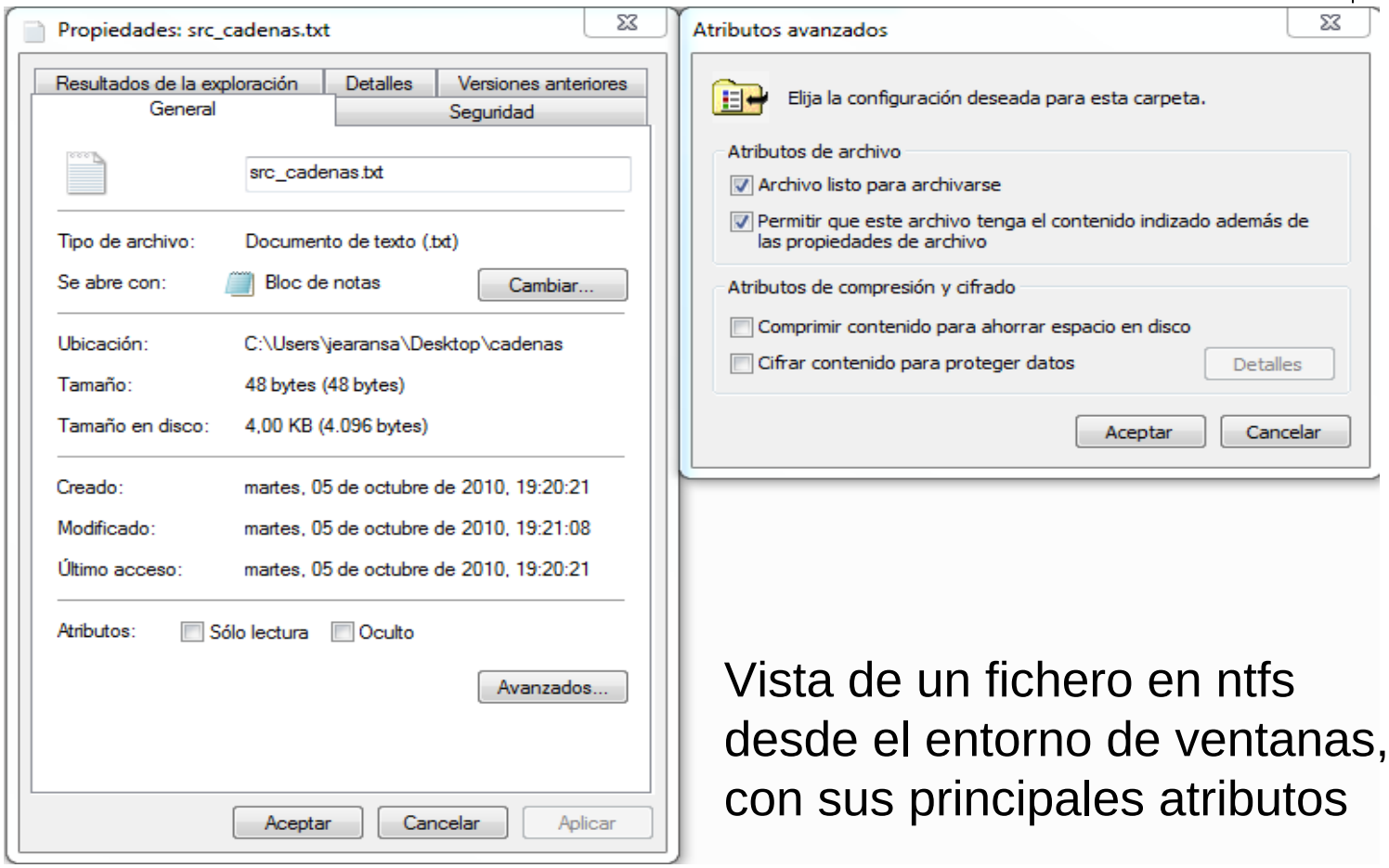
Se puede encontrar una lista completa de los atributos de los ficheros en NTFS en http://www.ntfs.com/ntfs-files-types.htm

3.3 Sistema de archivos ntfs



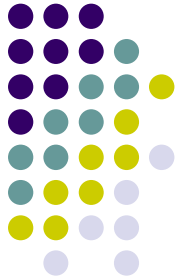
- Algunas características básicas sobre ntfs:
  - Longitud máxima de nombre de fichero: 255 caracteres
  - Tamaño máximo de partición: 256TB
  - Número máximo de ficheros: 2^32 – 1
  - Atributos disponibles:
    - Fechas de creación, modificación, cambio de atributos, acceso
    - Fichero de solo lectura, oculto, fichero de sistema, no indexar contenido, temporal, comprimido
  - Permisos soportados: ACL (no el mismo ACL que Unix, ver Sección 3.4)

3.3 Sistemas de archivos ntfs



Vista de un fichero en ntfs desde el entorno de ventanas, con sus principales atributos

3.4 Gestión de usuarios, grupos y permisos



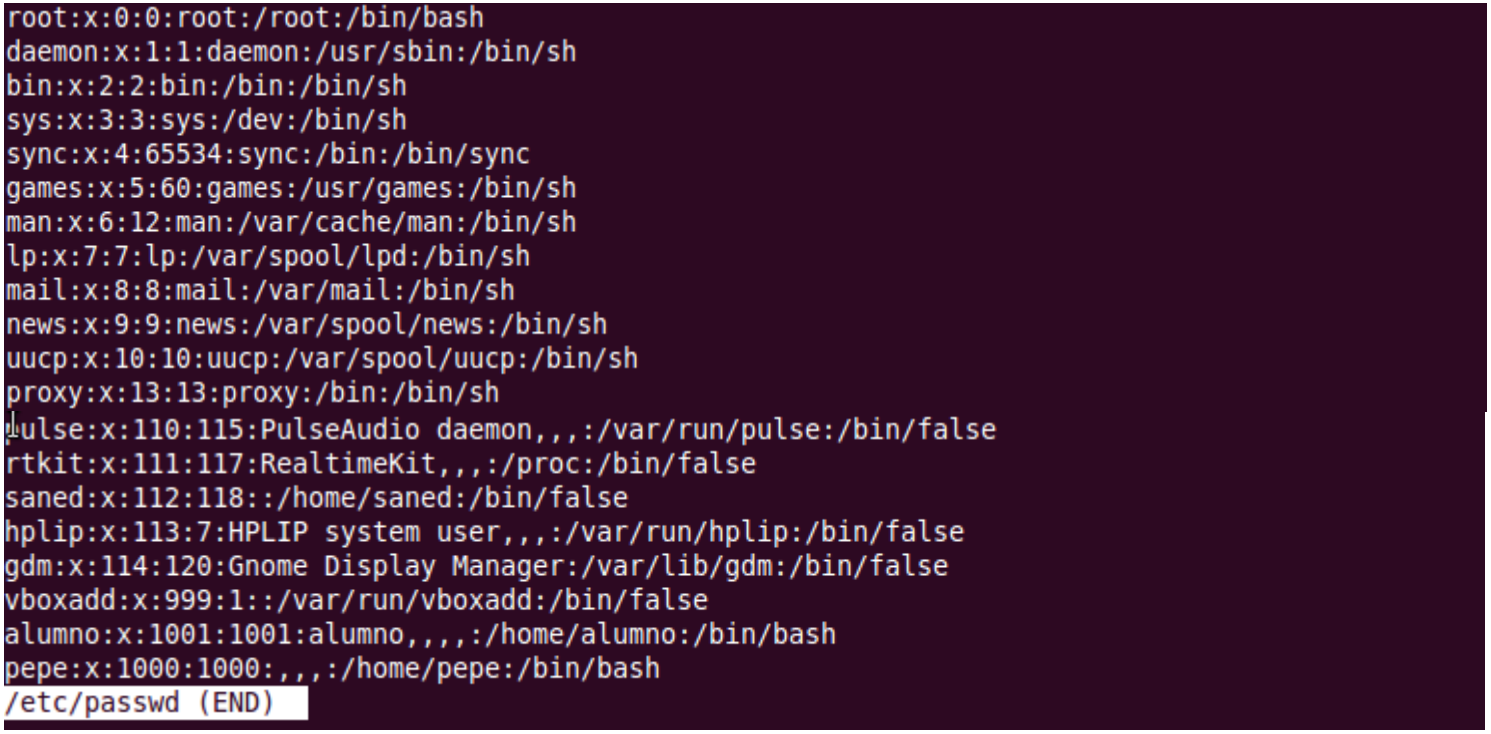
- Una de las principales funciones del sistema de archivos es permitir gestionar la información en nuestro ordenador desde el punto de vista de:
  - Quién puede acceder a ella
  - Quién puede modificarla
  - Quién puede ejecutarla
  - …

3.4 Gestión de usuarios, grupos y permisos

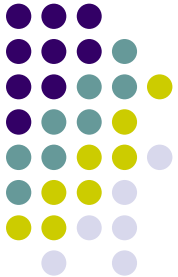
- Usuarios en Linux
  - Por defecto, durante la instalación, se genera una cuenta “root”; el cometido del mismo es administrar la máquina (crear usuarios, instalar/desinstalar aplicación, montar/desmontar dispositivos…)
  - Por seguridad, el mismo solo debe usarse para esos cometidos
  - Algunas distribuciones de Linux (como Ubuntu), y también Windows (desde XP) han “desactivado” el usuario “root”, y el primer usuario activado en una máquina puede actuar como tal (“sudo”, “ejecutar como Administrador”…)

3.4 Gestión de usuarios, grupos y permisos

- En las distro Linux podemos encontrar la lista de usuarios en el fichero “/etc/passwd”

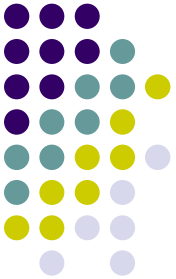


3.4 Gestión de usuarios, grupos y permisos



- Grupos en Linux
  - Por defecto, para cada usuario creado en una máquina se genera también un grupo
  - Por lo general, es más conveniente y cómodo asignar permisos a grupos, ya que de este modo, cuando creemos nuevos usuarios con permisos similares a los ya existentes, solo deberemos incluirlos en los grupos ya disponibles. Un ejemplo de esta “filosofía” de trabajo la da el propio sistema:
  - En las distro Linux suele haber una serie de grupos “por defecto” que representan algunas acciones típicas en el uso de la máquina; solo los miembros de esos grupos, a priori, podrán llevarlas a cabo:
    - Grupo lpr: hace uso de impresoras
    - Grupo cdrom: hacer uso de la unidad de cd
    - Grupo audio: hacer uso de los dispositivos de sonido
    - …

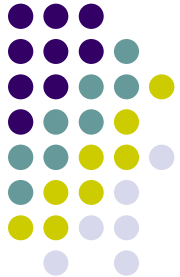
3.4 Gestión de usuarios, grupos y permisos



En general, en las distro Linux, podemos encontrar los grupos (y sus miembros) en el fichero “/etc/group”:

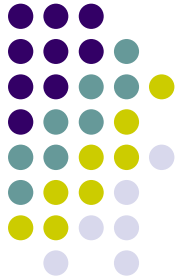
```
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:alumno
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:alumno
fax:x:21:alumno
voice:x:22:
cdrom:x:24:alumno
floppy:x:25:alumno
tape:x:26:alumno
sudo:x:27:
```

3.4 Gestión de usuarios, grupos y permisos



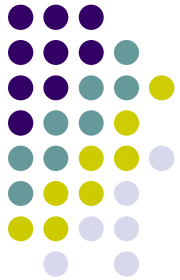
- Permisos en Linux:
  - Son una versión reducida de las ACL (listas de control de acceso), algunas veces llamado “POSIX.1e ACL”
  - Para cada entidad del sistema de archivos (ficheros o directorios) se dividen en **tres “grupos” de usuarios**:
    - El propietario o usuario (u)
    - El grupo al que está asignado el fichero (g); no tiene por qué ser igual al “grupo” de “u”
    - El resto de usuarios que pueden acceder a la máquina (o)
  - Para cada fichero o archivo distingue **tres posibles permisos**:
    - Lectura: r(ead), 4
    - Escritura: w(rite), 2
    - Ejecución: (e)x(ecute), 1

3.4 Gestión de usuarios, grupos y permisos



- El significado de r, w ó x depende de que nos encontremos ante un fichero o una carpeta
  - Ficheros
    - r ó 4: lectura sobre el fichero
    - w ó 2: edición del fichero
    - x ó 1: ejecución del fichero (incluso si el fichero no es ejecutable)
  - Carpetas
    - r ó 4: listar el contenido de un directorio (ls –l dir)
    - w ó 2: crear y borrar ficheros y directorios dentro de la carpeta
    - x ó 1: Acceso a un directorio (cd dir)

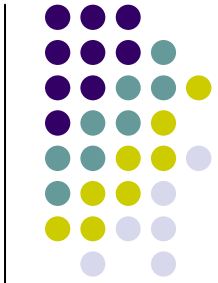
3.4 Gestión de usuarios, grupos y permisos



- Mandatos para la asignación de permisos
  - chmod ug+rw,o+r fichero (“+” añade permisos)
  - chmod a-x fichero (“a” abrevia a “todos”, “-” quita permisos)
  - chmod u=rw,o=r,a= fichero (“=“ asigna exactamente los permisos señalados, y los demás los quita)

- La sintaxis se compone de
  1. mandato “chmod”,
  2. seguido de la cadena de caracteres que representa los permisos que queremos asignar,
  3. y el fichero o carpeta correspondiente

3.4 Gestión de usuarios, grupos y permisos

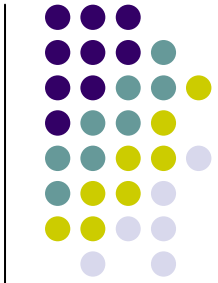


- Los permisos de un fichero o carpeta se pueden asignar por medio de tres números (propietario, grupo y otros), dados por la suma de los anteriores valores
  - chmod 652 fichero

    (ugo)

- El anterior mandato otorga permisos de
  - lectura y escritura al propietario o “user” (4 + 2 = 6),
  - lectura y ejecución al grupo o “group” (4 + 1 = 5),
  - escritura a otros u “others” (2)

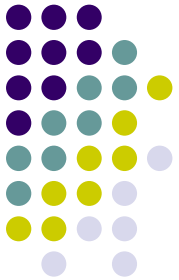
3.4 Gestión de usuarios, grupos y permisos



Algunas combinaciones numéricas habituales

- chmod 755 fichero (todos los permisos al propietario; grupo y otros sólo lectura y ejecución)
- chmod 777 fichero (todos los permisos para cualquier tipo de usuario)
- chmod 664 fichero (el fichero puede ser leído y modificado por el propietario y el grupo, sólo leído por el resto de usuarios, y no puede ser ejecutado)
- chmod 444 fichero (el fichero sólo es de lectura)
- chmod 555 fichero (sobre una carpeta, podemos abrirla y explorarla, pero no escribir en su interior)

3.4 Gestión de usuarios, grupos y permisos



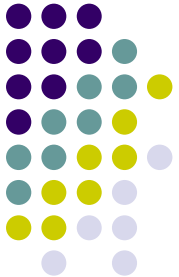
- Usuarios en “Windows NT”
  - Por defecto, durante la instalación, se genera una cuenta “Administrador”; el cometido del mismo es administrar la máquina (crear usuarios, instalar/desintalar aplicaciones, montar/desmontar dispositivos…)
  - Por seguridad, el mismo solo debe usarse para esos cometidos
  - Por defecto el mismo está desactivado; en ese caso, el SO obliga a que al menos un usuario pueda actuar como Administrador
  - Por defecto, el sistema también instala una cuenta de “Invitado”, usuario que se crea sin contraseña y que puede acceder a la máquina (aunque no pueda ejecutar…)

3.4 Gestión de usuarios, grupos y permisos

- Grupos en “Windows NT”
  - Por lo general, es más conveniente y cómodo asignar permisos a grupos, ya que de este modo, cuando creemos nuevos usuarios con permisos similares a los ya existentes, solo deberemos incluirlos en los grupos ya disponibles.
  - Suele haber una serie de grupos “por defecto” que representan algunas acciones típicas en el uso de la máquina; solo los miembros de esos grupos, a priori, podrán llevarlas a cabo:

Administradores	Los administradores tienen acceso completo y sin restricciones al equipo o dominio
Duplicadores	Pueden replicar archivos en un dominio
IIS_IUSRS	Grupo integrado usado por Internet Information Services.
Invitados	De forma predeterminada, los invitados tienen el mismo acceso que los miembros del grupo Usuarios, excepto la cuenta de invitado ...
Lectores del registro de eventos	Los miembros de este grupo pueden leer registros de eventos del equipo local.
Operadores criptográficos	Los miembros tienen autorización para realizar operaciones criptográficas.
Operadores de configuración de red	Los miembros en este equipo pueden tener algunos privilegios administrativos para administrar la configuración de las característica...
Operadores de copia de seguridad	Los operadores de copia de seguridad pueden invalidar restricciones de seguridad con el único propósito de hacer copias de segurid...
Usuarios	Los usuarios no pueden hacer cambios accidentales o intencionados en el sistema y pueden ejecutar la mayoría de aplicaciones
Usuarios avanzados	Los usuarios avanzados se incluyen para la compatibilidad con versiones anteriores y poseen derechos administrativos limitados
Usuarios COM distribuidos	Los miembros pueden iniciar, activar y usar objetos de COM distribuido en este equipo.
Usuarios de escritorio remoto	A los miembros de este grupo se les concede el derecho de iniciar sesión remotamente
Usuarios del monitor de sistema	Los miembros de este grupo tienen acceso a los datos del contador de rendimiento de forma local y remota
Usuarios del registro de rendimiento	Los miembros de este grupo pueden programar contadores de registro y rendimiento, habilitar proveedores de seguimiento y recopil...
HomeUsers	HomeUsers Security Group
Usuarios del depurador	Los usuarios del depurador pueden depurar procesos en este equipo, con carácter local y remoto
__vmware__	VMware User Group

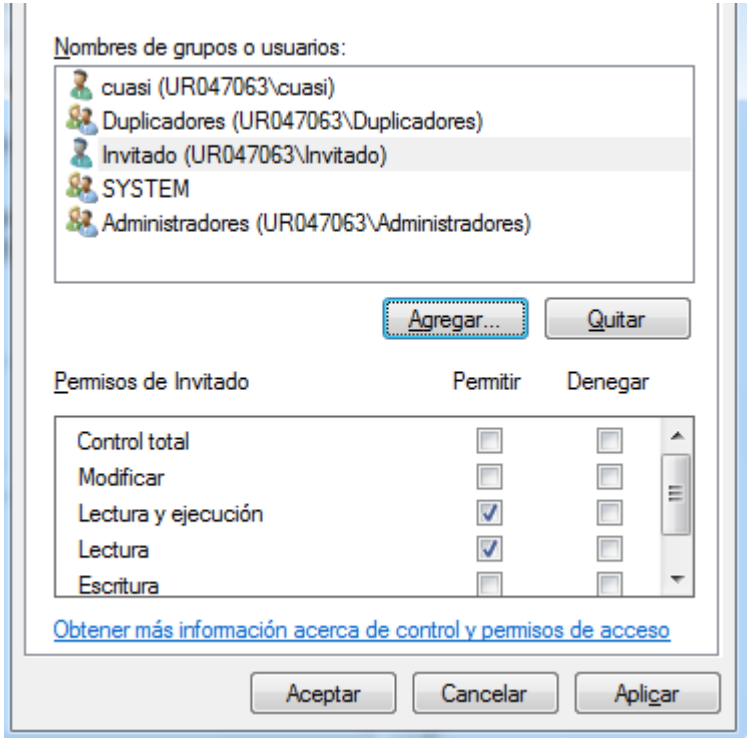
3.4 Gestión de usuarios, grupos y permisos



- Permisos en “Windows NT”
  - Están basados también en listas de control de acceso (como en Linux), pero son capaces de distinguir más “ACE’s” (entradas de control de acceso) que las de Linux. Son muy similares a las ACL’s de NFSv4 (sistema de ficheros en red detallado en el RFC 3530)
  - En particular, cada entidad del sistema (ficheros, directorios, gestor de usuarios, periféricos, servicios de red…) puede tener permisos independientes para cualquier grupo o usuario de la máquina (no solo “ugo”)
  - Cada fichero o carpeta puede tener una lista de permisos bastante más extensa de lo posible en las POSIX 1e.ACL (“rwx”)

3.4 Gestión de usuarios, grupos y permisos

- Cualquier usuario o grupo es susceptible de tener sus propios permisos sobre un fichero o carpeta



3.4 Gestión de usuarios, grupos y permisos

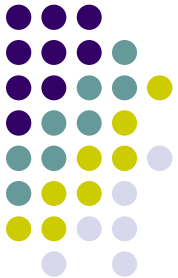
La lista de permisos de acceso (access rights) para cualquier objeto (ACE) incluye tres categorías

- Permisos genéricos
- Permisos de acceso estándar
- Permisos específicos de objeto

GR → Generic_Read
GW → Generic_Write
GE → Generic_Execute
GA → Generic_ALL
AS → Right to access SACL

Entre estos objetos se incluyen, por supuesto, ficheros y carpetas, pero también objetos genéricos (impresoras, servicios, como mail o ftp, usuarios, grupos…)

3.4 Gestión de usuarios, grupos y permisos



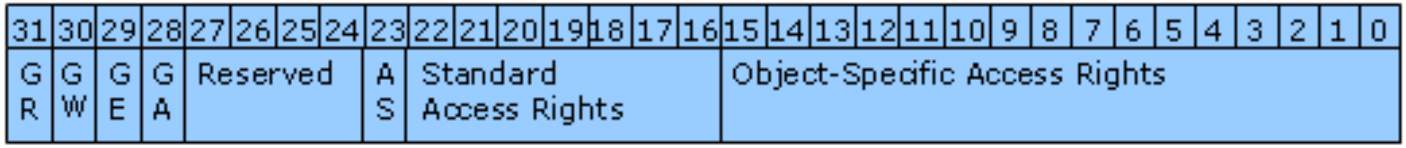
- Los permisos genéricos (GR, GW, GE, GA) se aplican a cualquier objeto, pero su significado puede variar
- Si se asignan, se hacen corresponder con algunos de los permisos “estándar” o “específicos de objeto”
- Sirven de “atajo” para asignar permisos más cómodamente

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GR	GW	GE	GA	Reserved				AS	Standard Access Rights							Object-Specific Access Rights															

Key
GR Generic Read
GW Generic Write
GE Generic Execute
GA Generic All
AS Right to access SACL

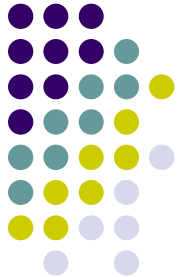
3.4 Gestión de usuarios, grupos y permisos

- Permisos de acceso estándar (aplicables también a todo objeto)
  - Delete: permiso para borrar
  - Read_control: lectura de la descripción de seguridad del objeto
  - Synchronize: entornos con múltiples máquinas
  - Write_dac: modificar listas de control de acceso discrecional (para usuarios y grupos concretos)
  - Write_owner: modificar el propietario



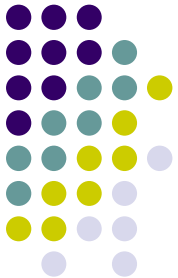
Key
GR Generic Read
GW Generic Write
GE Generic Execute
GA Generic All
AS Right to access SACL

3.4 Gestión de usuarios, grupos y permisos



- Permisos (ACEs) específicos para objetos
- **En el caso de ficheros y carpetas**, contiene al menos los siguientes:
  - Traverse folders/execute files
  - List folder/read data
  - Read attributes (los propios de ntfs)
  - Read extended attributes (propios de otros programas)
  - Create files/write data
  - Create files/append data
  - Write attributes
  - Write extended attributes
  - Delete subfolders and files
  - Delete
  - Read permissions
  - Change permissions
  - Take ownership
  - Synchronize

3.4 Gestión de usuarios, grupos y permisos

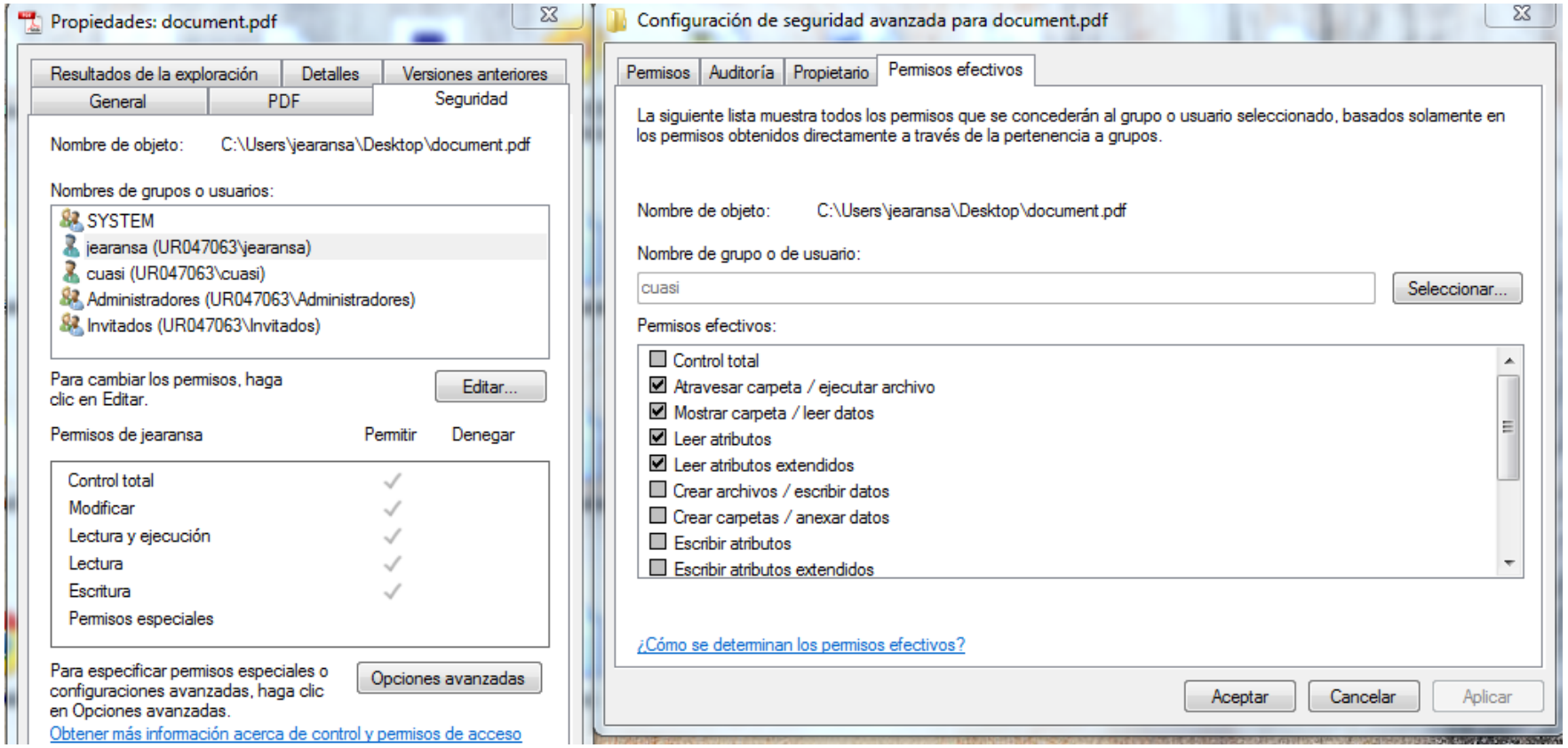


- Permisos específicos para objetos
- **En el caso de servicios**, contiene al menos los siguientes (observar que son distintos de los de ficheros y carpetas):

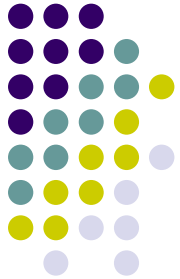
  - Full control
  - Query template
  - Change template
  - Query status
  - Enumerate dependents
  - Start
  - Stop
  - Pause and continue
  - Interrogate
  - User defined control
  - Delete
  - Read permissions
  - Change permissions
  - Take ownership
  - Synchronize

3.4 Gestión de usuarios, grupos y permisos

Interfaz en “Windows NT” para modificar estos permisos

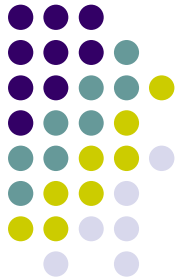


3.5 Tareas o procesos y servicios



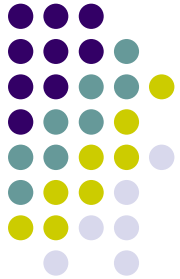
- Una de las componentes y funciones principales y más complejas de un SO es la gestión de procesos
- La misma involucra, de una forma u otra, gestión de memoria, de E/S, de ficheros y de sincronización entre distintos procesos
- Asimismo, utiliza recursos tales como el procesador, la memoria principal, la virtual, los periféricos (dispositivos de E/S)…

3.5 Tareas o procesos y servicios



- Un **proceso** es un programa que se encuentra en ejecución
- Los **programas** son conjuntos de órdenes en lenguaje máquina alojadas en un dispositivo de almacenamiento
- Un mismo programa, cuando es ejecutado varias veces, da lugar a varios procesos

3.5 Tareas o procesos y servicios



- Jerarquía de procesos:
  - Los procesos en ejecución en un SO forman una estructura de “árbol” en la que cada proceso dispone de un proceso “padre”, creando una jerarquía entre los mismos
  - En Unix se mantiene la información de esa jerarquía, cada proceso conoce a su antecesor

3.5 Tareas o procesos y servicios

Resultado de ejecutar pstree en una shell de Linux

3.5 Tareas o procesos y servicios

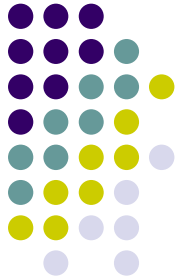
- Asimismo, cada proceso guarda información sobre ciertas variables de entorno que se crean cuando el mismo se ejecuta

```
USERNAME=alumno
GDM_LANG=es_ES.UTF-8
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
DISPLAY=:0.0
LANG=es_ES.UTF-8
XAUTHORITY=/var/run/gdm/auth-for-alumno-cGYMHU/database
SHELL=/bin/bash
GDMSESSION=gnome
SPEECHD_PORT=7561
PWD=/home/alumno
```

Aspecto del fichero “environ” en /proc de un proceso cualquiera en linux

- ¿Dónde se mantiene esta información?...

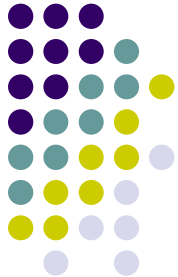
3.5 Tareas o procesos y servicios



- Información de los procesos; cada proceso en ejecución da lugar a elementos de información que se pueden agrupar en tres categorías

1. Información del BCP (bloque de control de proceso): en la tabla de control de procesos
2. Estado del procesador: en el procesador, cuando el proceso está en ejecución; si no, en el mapa de memoria
3. Imagen o mapa de memoria del proceso: en la memoria principal (o RAM) o virtual (swap)

3.5 Tareas o procesos y servicios
Bloque de Control de Proceso

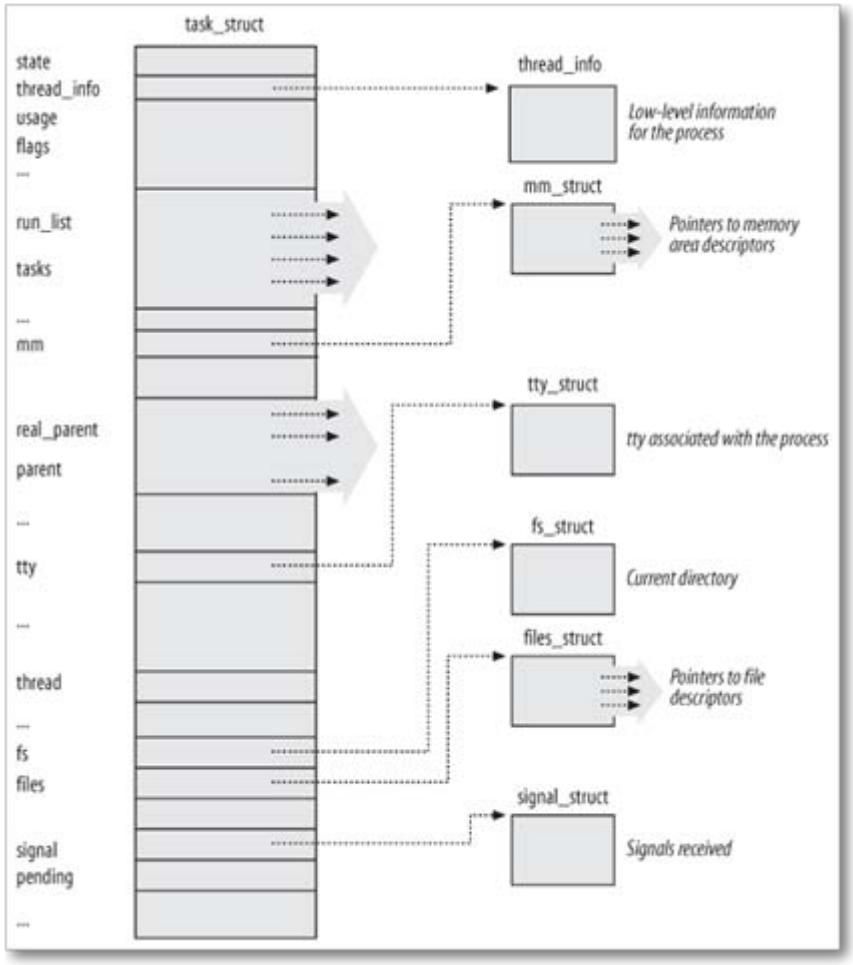


- La **tabla de control de procesos** está formada por bloques que representan cada proceso en ejecución;
- Cada proceso en ejecución dispone de un bloque de control de proceso (BCP) que contiene información sobre:
  - Identificador del proceso y del “padre”
  - Identificador de usuario – grupo
  - Estado del procesador
  - Información de control del proceso:
    - Estado del proceso
    - Prioridad
    - Información de planificación
    - Regiones de memoria asignadas al proceso
    - Ficheros abiertos
  - …

3.5 Tareas o procesos y servicios

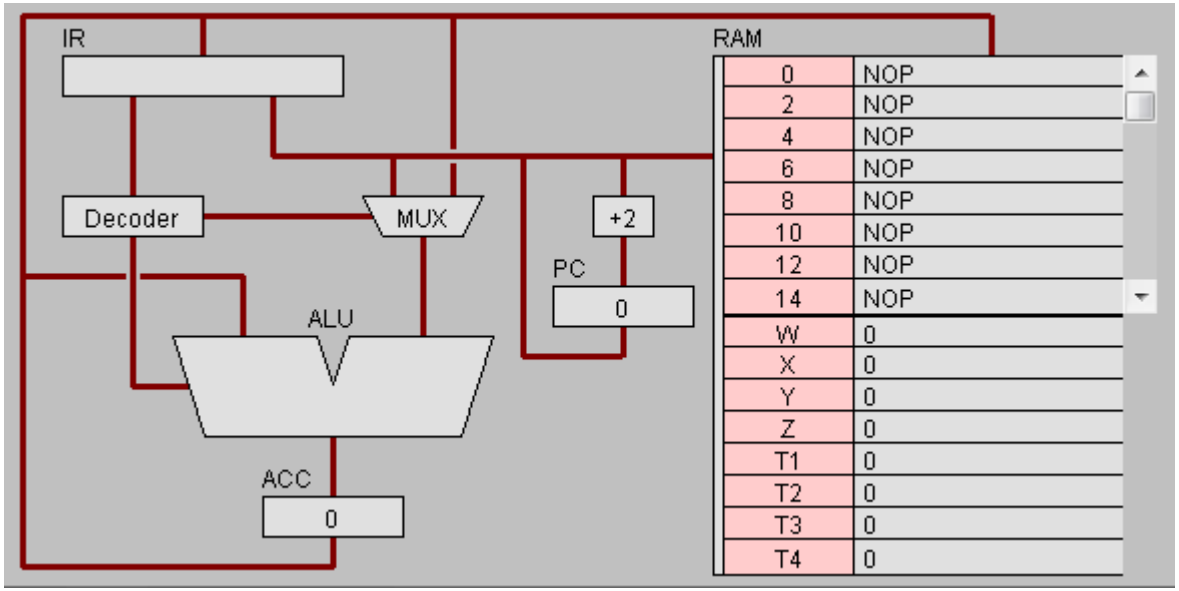
Aspecto del BCP de un proceso en Linux; el mismo contiene:

- información “estática” del proceso: usuario, identificador, proceso padre
- información “dinámica”: valores del procesador en el momento de inicio del proceso, o cuando es “expulsado” del procesador (pausado, suspendido…)

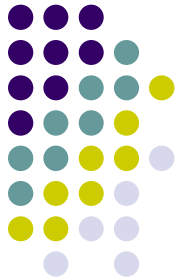


3.5 Tareas o procesos y servicios
Estado del procesador

- Cuando un proceso se encuentra en ejecución, hace uso de los recursos que el procesador facilita (contador de programa, acumulador, registros auxiliares…)



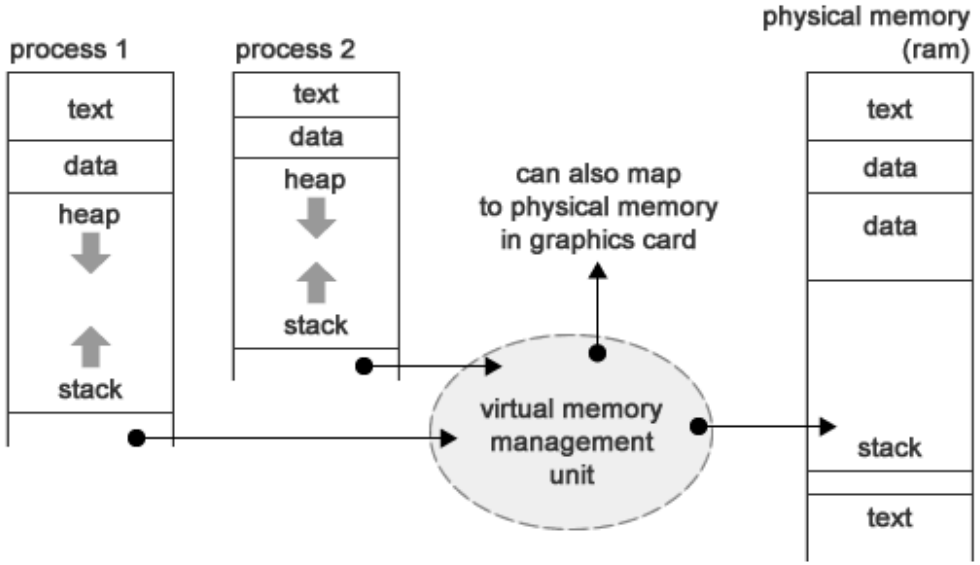
3.5 Tareas o procesos y servicios



- Cuando el proceso deja de estar en ejecución (por ejemplo, porque está a la espera de que termine una operación de E/S, o porque hay otro con mayor prioridad) el estado de los registros, contador de programa… se almacena en el BCP
- Cuando el programa vuelve a ser ejecutado por el procesador, se lee su información del BCP, se escribe a la UCP, y se continúa con la ejecución

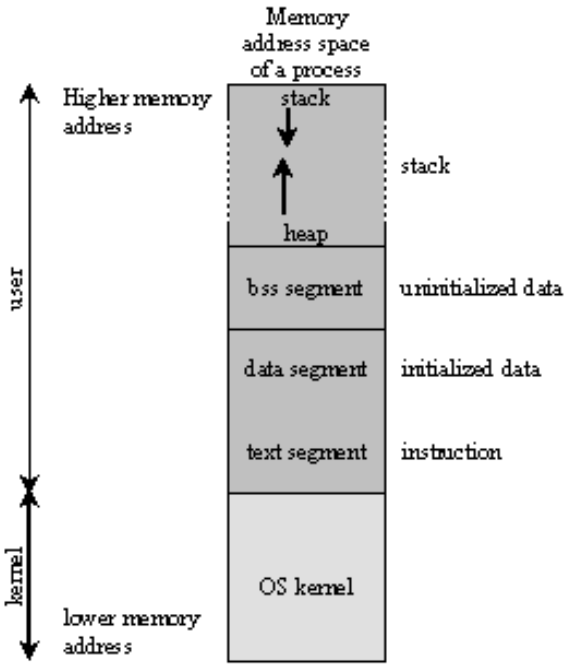
3.5 Tareas o procesos y servicios
Imagen o mapa de memoria del proceso

- Viene formada por los espacios de memoria principal (o RAM) que un proceso puede ocupar
- En los mismos se aloja información sobre
  - Texto del proceso
  - Datos del proceso
  - Pila del proceso

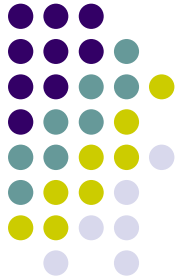


3.5 Tareas o procesos y servicios

- Texto del proceso: formado por el **código máquina** del programa; se considera estático (de tamaño fijo), y sobre el mismo sólo se hacen operaciones de lectura o ejecución
- Datos:
  - Data segment: Alojado estáticamente y con valores globales para todo el proceso
  - BSS segment: Alojados estáticamente e inicializados a cero por defecto
- Pila:
  - Heap (montículo): provee al proceso de memoria dinámica, se gestiona por punteros
  - Stack (pila): en ella nos encontramos los parámetros de las llamadas a subrutinas o las variables locales. También es de tamaño variable

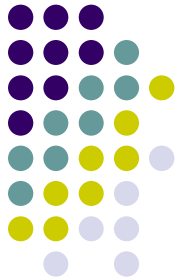


3.5 Tareas o procesos y servicios



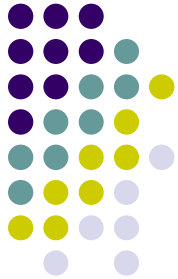
- Multitarea, activación e interrupción de procesos
  - La multitarea permite activar un proceso, detectar que el mismo se encuentra realizando una operación que permite “bloquearlo” (por ejemplo, lectura/escritura a un dispositivo externo, espera de una conexión a Internet…) y activar otro distinto en ese intervalo
  - La activación de un proceso requiere recuperar el mismo (el estado de sus registros) desde su BCP y proseguir con la ejecución de acuerdo a su mapa de memoria
  - La interrupción requiere copiar desde el procesador toda la información al BCP y conservar su estado

3.5 Tareas o procesos y servicios



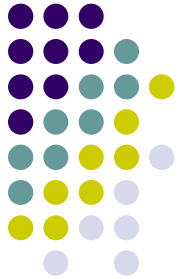
- De la noción de multitarea, aparecen los siguientes estados para procesos:
  - En ejecución: ocupa el procesador
  - Bloqueado: a la espera de un evento necesario para la ejecución del proceso
  - Listo: preparado para ser ejecutado (aquí entran en juego las prioridades)
- Existen algunos otros estados:
  - Suspendido: el proceso se encuentra en la memoria virtual (o swap) para reducir la carga de multitarea y liberar memoria principal

3.5 Tareas o procesos y servicios



- El SO suele facilitar servicios que permiten:
  - Crear procesos; por ejemplo, en Unix, el servicio “fork” crea un proceso similar a su proceso “padre”
  - Ejecutar procesos:
    - En modo de lotes o batch: sin estar vinculado a ninguna terminal (la salida y entrada del mismo se hace de forma “oculta” al usuario)
    - En modo interactivo: desde una terminal desde o a la que lanza información (en primer plano o en segundo plano)
  - Terminar procesos: porque el procesos ha terminado, porque ha producido un error…
  - Cambiar el ejecutable: exec en Linux
  - Enviar señales a procesos: kill –s “SIGNAL” “PID”

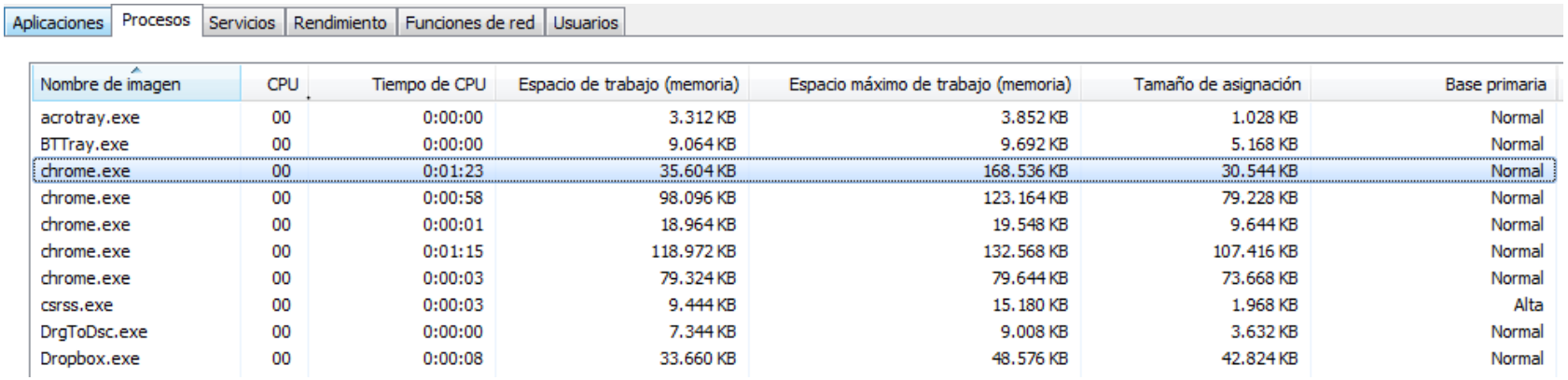
3.6 Gestión de memoria



- El gestor de memoria se encarga de gestionar la memoria principal (y virtual) del ordenador para que la misma pueda ser compartida por los distintos procesos en ejecución
- Entre otras múltiples tareas, debe asegurar que cada proceso cuente con un espacio de memoria independiente
- Está fuertemente ligado al hardware (tipo y tamaño de memoria principal) subyacente

3.6 Gestión de memoria

- La gestión de memoria está íntimamente ligada a la gestión de procesos, ambas tienen el fin de permitir que el ordenador puede ejecutar los procesos en modo de multitarea

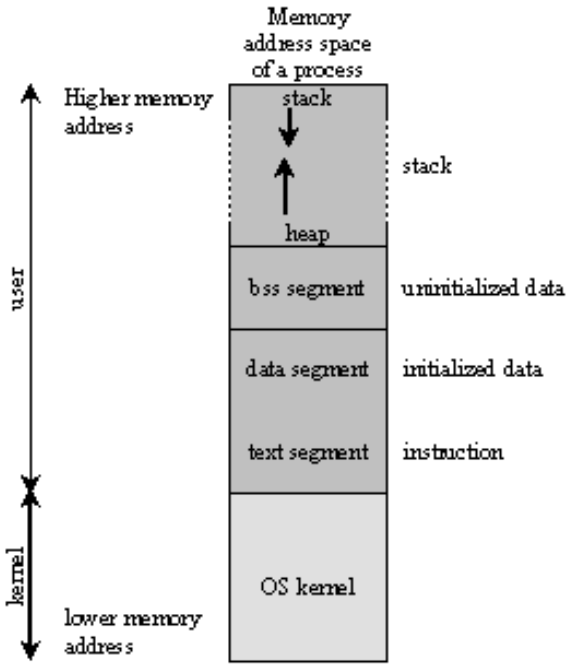


Nombre de imagen	CPU	Tiempo de CPU	Espacio de trabajo (memoria)	Espacio máximo de trabajo (memoria)	Tamaño de asignación	Base primaria
acrotray.exe	00	0:00:00	3.312 KB	3.852 KB	1.028 KB	Normal
BTTray.exe	00	0:00:00	9.064 KB	9.692 KB	5.168 KB	Normal
chrome.exe	00	0:01:23	35.604 KB	168.536 KB	30.544 KB	Normal
chrome.exe	00	0:00:58	98.096 KB	123.164 KB	79.228 KB	Normal
chrome.exe	00	0:00:01	18.964 KB	19.548 KB	9.644 KB	Normal
chrome.exe	00	0:01:15	118.972 KB	132.568 KB	107.416 KB	Normal
chrome.exe	00	0:00:03	79.324 KB	79.644 KB	73.668 KB	Normal
csrss.exe	00	0:00:03	9.444 KB	15.180 KB	1.968 KB	Alta
DrgToDsc.exe	00	0:00:00	7.344 KB	9.008 KB	3.632 KB	Normal
Dropbox.exe	00	0:00:08	33.660 KB	48.576 KB	42.824 KB	Normal

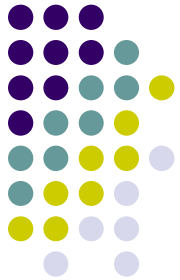
3.6 Gestión de memoria

En la gestión de memoria podemos distinguir tres niveles:

- Nivel de procesos: decide cuánta y qué parte de la memoria principal es ocupada por un proceso
- Nivel de regiones: para cada proceso, decide qué partes se asignan a “texto”, “datos” y “pila”
- Nivel de zonas: dentro de la pila, puede haber distintas políticas de gestión de la memoria asignada

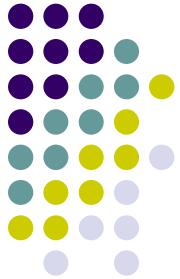


3.6 Gestión de memoria



- El gestor de memoria se encarga en cada uno de los niveles de
  - Nivel de procesos
    - Crear los mapas de memoria
    - Eliminar los mapas de memoria
    - Duplicar los mapas de memoria (fork en Linux)
  - Nivel de regiones
    - Crear regiones (texto, data…) dentro del mapa de un proceso
    - Eliminar regiones
    - Cambiar el tamaño de regiones (“heap” o “stack”)
    - Duplicar una región en otro mapa (fork)
  - Nivel de zonas
    - Reservar zonas (“malloc” o “new” en C++) en el heap o el stack
    - Liberar zonas (“delete” o “free” en C++)
    - Cambiar el tamaño de zonas

3.6 Gestión de memoria



- Cada una de las operaciones anteriores requiere de una algorítmica compleja para optimizar los espacios asignados a cada proceso
- Una de esas múltiples técnicas es la **paginación,** proceso por el cual la memoria virtual se “divide” en segmentos iguales donde se ubican “segmentos” de los procesos, evitando la fragmentación de la memoria

Repaso a las principales partes del SO