

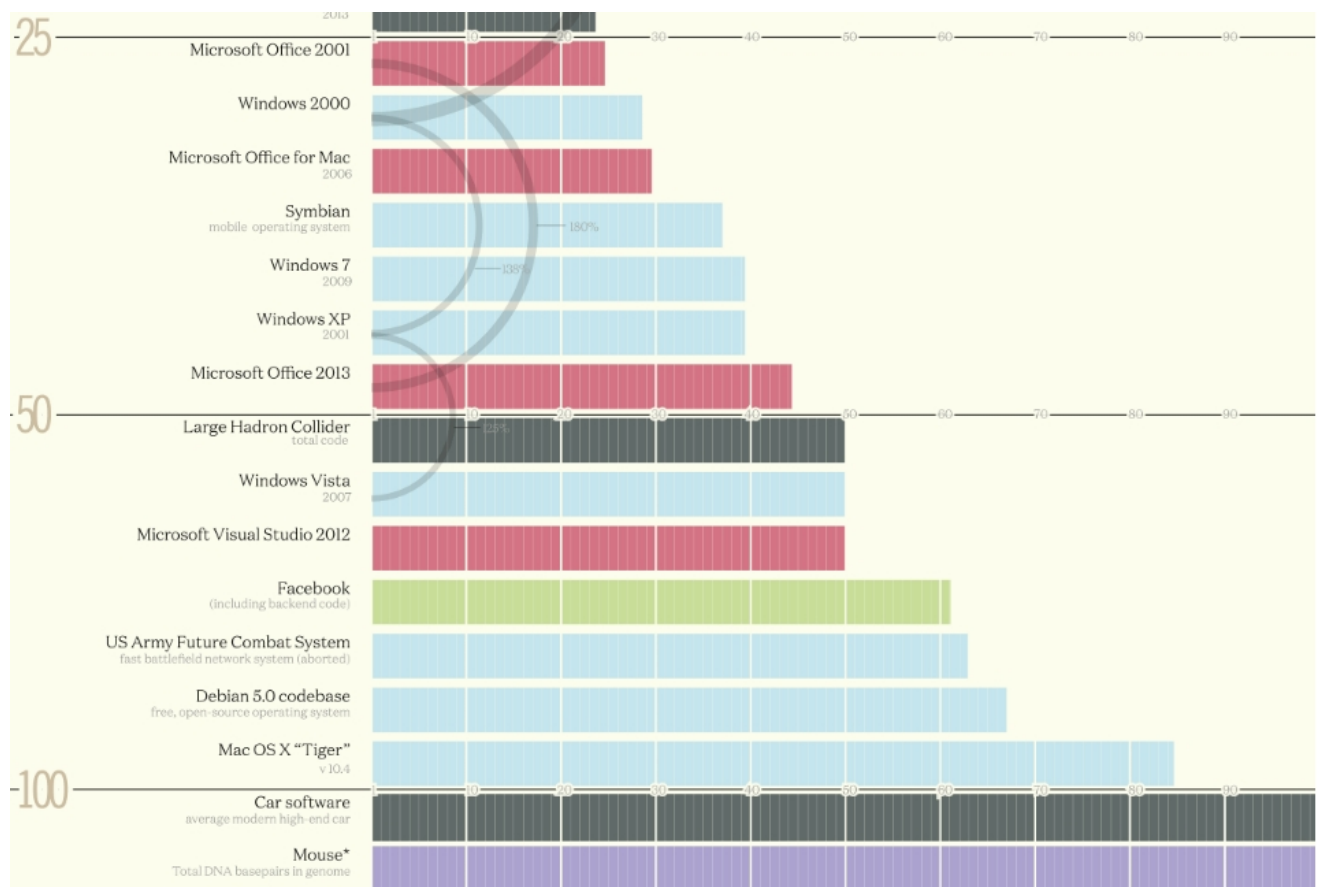
Introducción

Sistemas Operativos

Conceptualmente en el seno de las Ciencias de la Computación lo Sistemas Operativos ocupan un lugar destacado.

Por un lado, porque los **conceptos** que se manejan en el **ámbito de un sistema operativo** están categorizados cómo algunos de los **más complejos** en la informática.

Para tener una idea, un Sistema Operativo moderno está compuesto por unas 50 millones de líneas de código fuente.



Alguien en este momento está escribiendo algunas líneas de código para algún sistema operativo que necesariamente no tiene que ser para una PC.

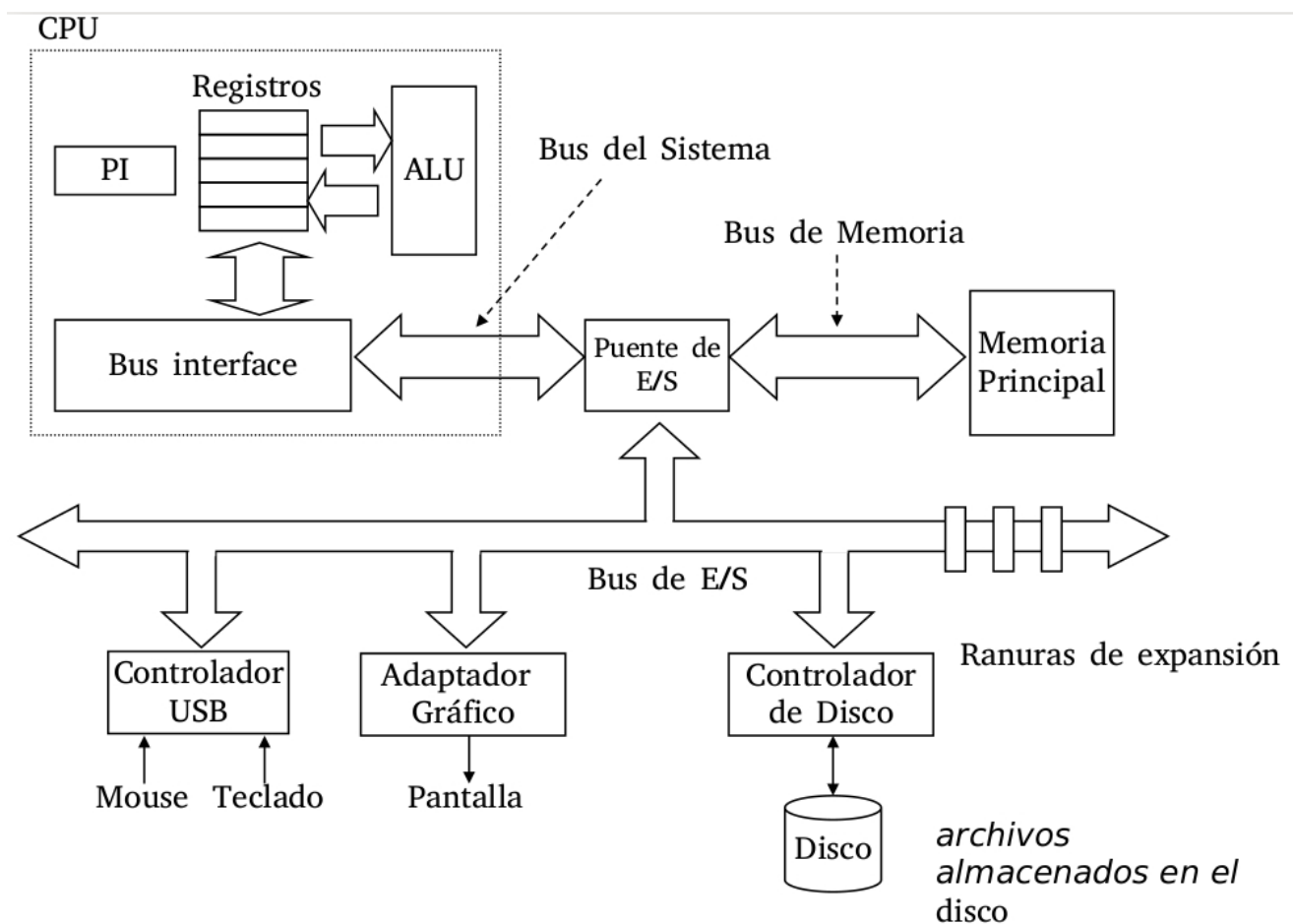
Por otro lado, los **conceptos manejados en los sistemas operativos están dentro de la gama de los más utilizados en las Ciencias de la Computación y de hecho, muchos de ellos les van a parecer familiares y de la vida cotidiana:**

- Tratar de hacer dos cosas al mismo tiempo.
- Haberse equivocado de cola en el supermercado.
- Tratar de que tu hermano, con el cual compartís la habitación, no se meta con tus cosas.

entre otras.

Una Computadora

Puede parecer casi trivial pero para poder dominar los conceptos que se manejan en sistemas operativos se debe tener bien en claro: ¿Qué es una Computadora? y además, Cuales son sus partes:



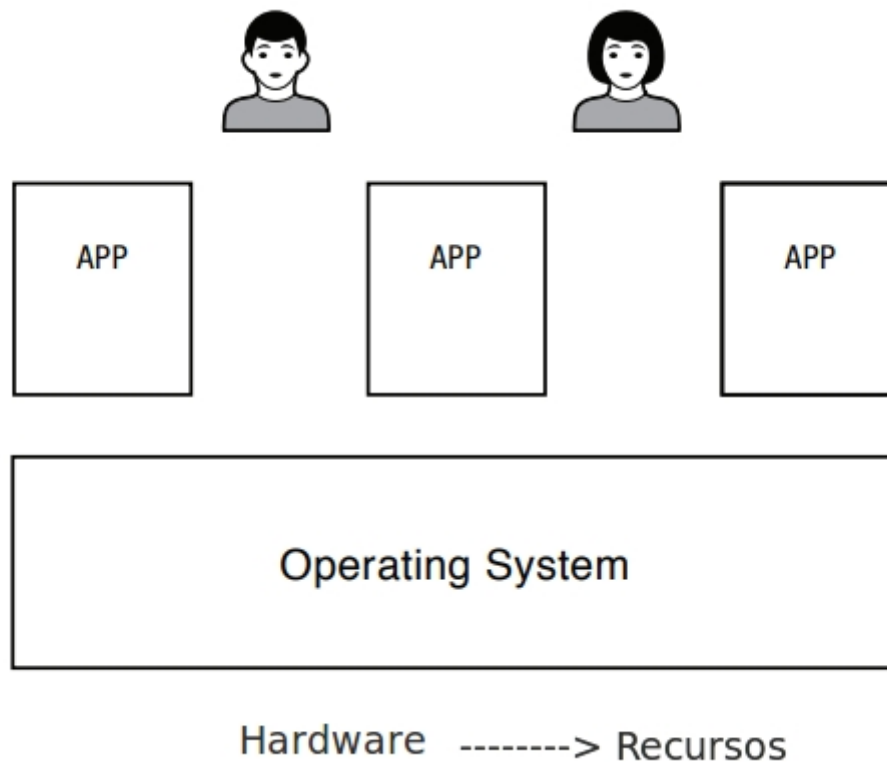
Las partes esenciales como todos sabemos son :

- CPU: Central Process Unit.
- Main Memory: Memoria Principal.
- Devices: Dispositivos de Entrada o Salida.
- Buses: conexiones.

(ver: BRY Chap:1 pag:8)

¿Qué es un Sistema Operativo?

Un Sistema Operativo (OS) es la capa de software que maneja los recursos de una computadora para sus usuarios y sus aplicaciones. [DAH]



Según Commers[COM]:

Un sistema operativo está diseñado para ocultar detalles de hardware de bajo nivel y para crear una máquina abstracta que le proporcione a las aplicaciones servicios de alto nivel.

Según Maurice Bach [BCH]:

El sistema operativo interactúa directamente con el hardware, proporcionando servicios comunes a los programas y aislándolos de idiosincrasias de hardware.

otra definición dada por [STV] :

Un Sistema Operativo puede definirse como el software que controla los recursos de hardware de una computadora y provee un ambiente bajo el cual los programas pueden ejecutarse.

Normalmente los sistemas operativos son invisibles a los ojos de sus usuarios por ejemplo alguien le prestó atención al sistema operativo de:

- Una consola de video juegos
- Un teléfono
- Un coche
- Un avión

- Una Heladera
- Un karaoke
- Una cámara de fotos

De todas formas en este curso se estudiará a fondo a los sistemas operativos de propósito general, muchos de uds. ya han interactuado con alguno como por ejemplo Linux, Mac OS o Windows.

Hoy en día cualquier persona que utilice una computadora debe lidiar, aun inconscientemente, con un sistema operativo.

En un sistema operativo de propósito general, los usuarios interactúan con aplicaciones, estas aplicaciones se ejecutan en un ambiente que es proporcionado por el sistema operativo. A su vez el sistema operativo hace de mediador para tener acceso al hardware del equipo.

Un Poco de Historia

Una cosa curiosa que se puede ver a lo largo de la historia de los sistemas operativos es que éstos evolucionaron a la par de la arquitectura de las computadoras que los ejecutaban. Según Tanenbaum, las generaciones de computadoras están ligadas a las generaciones de sistemas operativos, para él, este matcheo no falla:

Generación de Tubos de Vacío y tableros Enchufables o Primera Generación (1945-1955)

La construcción de computadoras digitales tuvo su auge hacia fines de la Segunda Guerra Mundial. Hacia fines de los 40 ya había varios grupos de investigación trabajando en la construcción de computadoras digitales de tubos de vacío:

- en Princeton John Von Neuman (ENIAC).
- en Harvard Howard Aiken.
- en Alemania Konrad Zuse.



Baby Manchester [⁴].

Todas estas máquinas utilizaban tubos de vacío y tableros enchufables para ser programadas. Literalmente se podía decir que los programas estaban atados con alambres. Estas computadoras eran enormes, tan grandes que a veces conformaban edificios. Los programas se escribían en lenguaje de máquina absolutos.

18/7/48

Kilburn Highest Factor Routine (amended)-

Instruction	C	25	26	27	Line	01234	131415
-24 to C	$-G_1$	-	-	-	1	00011	010
c to 26			$-G_1$		2	01011	110
-26 to C	G_1				3	01011	010
c to 27			$-G_1$	G_1	4	11011	110
-23 to C	a	T_{n-1}	$-G_N$	G_N	5	11101	010
sub 27	$a - G_N$				6	11011	001
Test					7	-	011
Add 20 to bl.					8	00101	100
sub. 26	T_N				9	01011	001
c to 25		T_N			10	10011	110
-25 to C					11	10011	010
Test					12	-	011
stop	0	0	$-G_N$	G_N	13		111
-26 to C	G_N	T_N	$-G_N$	G_N	14	01011	010
sub. 21	G_{N+1}				15	10101	001
c to 27	G_{N+1}			G_{N+1}	16	11011	110
-27 to C	$-G_{N+1}$				17	11011	010
c to 26			$-G_{N+1}$		18	01011	110
22 to bl.		T_N	$-G_{N+1}$	G_{N+1}	19	01101	000

or 000

20	-3	10111 etc
21	1	10000
22	4	00100

23	-a
24	G_1

	init.	final
25	-	$T_N (\neq 0)$
26	-	$-G_N$
27	-	G_N

or 10100

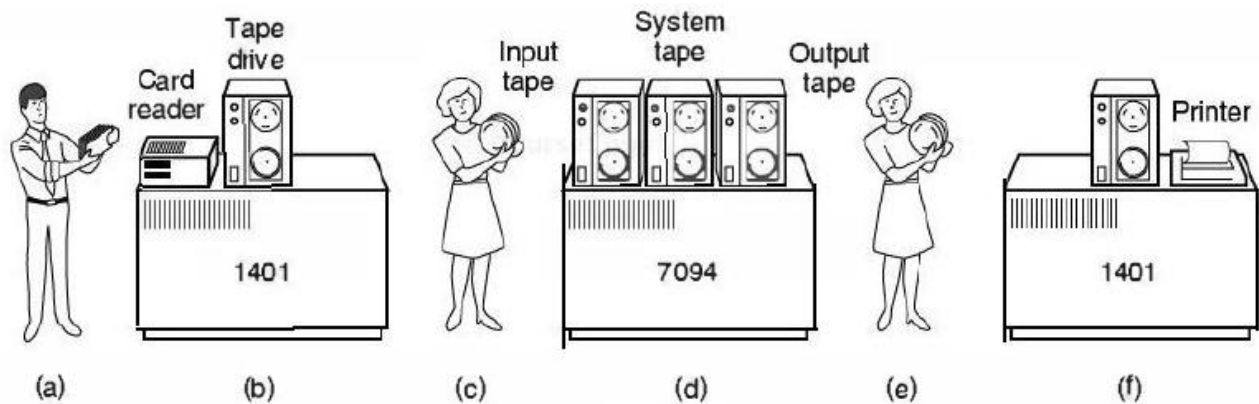
Generación de Transistores y de sistemas por lotes o Segunda Generación (1955-1965)

Esta es la generación, donde surgen las tarjetas perforadas. En esta época había claras distinciones entre diseñadores, programadores, operadores, armadores y personal de mantenimiento.

La idea principal en esta época era que debido a que las computadoras eran muy caras, y pocas empresas / organizaciones tenían acceso a ellas, los programadores:

1. escribían su trabajo (job) en papel,
2. perforaban sus tarjetas,

3. llevaban las mismas al cuarto de operaciones y se lo entregaban al operador.
4. Cuando uno de estos trabajos terminaba normalmente el operador dirigía la salida a la impresora.
5. Dicha impresión se llevaba al cuarto de impresiones para ser devuelto al programador.



An early batch system. (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape. (c) Operator carries input tape to 7094. (d) 7094 does computing. (e) Operator carries output tape to 1401. (f) 1401 prints output.

Generación de Circuitos Integrados y multiprogramación o Tercera Generación (1965-1980)

Hacia la década de los 60 existían dos líneas de computadoras muy distintas en lo que era su arquitectura, una orientada al procesamiento de datos bancarios la IBM-1401:



y la otra al procesamiento de cálculo numérico, la IBM-7094: mostrar video



ambas con características muy distintas. La idea de los diseñadores de computadoras era tender a la unificación.

Para ello IBM lanzó su producto estrella la IBM-360. Este producto era una línea de computadoras arquitecturalmente compatibles. Justamente en esta características estaba su fortaleza y a su vez su debilidad. Además, fue la primera serie de computadoras construida con circuitos integrados.




Esta generación de computadoras introdujo conceptos técnicos muy importantes para los sistemas operativo. En los sistemas operativos anteriores cuando un programa tenía que realizar una operación de E/S, la CPU permanecía osiosa hasta que se completaba dicha operación. Por el contrario los programas que ejecutaban cálculo numérico tienen muy poco uso de E/S.

[TAN] : "... La solución a la que se llegó fue dividir la memoria en varias secciones, con un trabajo distinto en cada partición, ... Mientras un trabajo estaba esperando que terminara su E/S, otro podía estar usando la CPU. Si se podían tener en la memoria principal suficientes trabajos a la vez, la CPU podía mantenerse ocupada casi todo el tiempo. Tener múltiples trabajos en la memoria a la vez requiere hardware especial para proteger cada trabajo contra espionaje o por parte de los demás, pero la 360 y otros sistemas de tercera generación estaban equipados con este hardware. ..."

Otra técnica que surgió en esta generación de sistemas operativos es la llamada técnica de spoolin (*Simultaneous Peripheral Operation On Line*) que permite manejar varios dispositivos simultáneamente.

Aun así los sistemas operativos de tercera generación eran lentos, el tiempo de espera entre la entrega de un trabajo y su resultado podían ser varias horas.

En esta generación surge una técnica derivada de la multiprogramación llamada **time Sharing**  [corbato](#).

Básicamente la idea es la de transformar una computadora en un sistema multiusuario con tiempo de respuesta muy cortos, de forma tal, que cada usuario creyera que estaba usando su propia computadora. Esto es debido a que estadísticamente los usuarios de una computadora no realizan operaciones de calculo o entrada/salida simultáneamente, por ende la idea es compartir el tiempo de uso con los demás usuarios. Algunos estarán charlando entre ellos, otros tomando café, otros no están logeados, otro estará haciendo debugging, etc.

Esta idea se plasmó en **MULTICS**, un sistema operativo que no tuvo mucho éxito y fue discontinuado por sus creadores. Pero **MULTICS** propuso varias técnicas innovadoras que aportaron en el diseño de sistemas operativos. De hecho hay ciertas reminiscencias de **MULTICS** en **UNIX** [linux2multics](#)

Hacia fines de los 60, el desarrollo de las minicomputadoras hizo posible que su costo se redujera, una PDP-11 costaba unos 120000 dolares un 5% de sus predecesoras (IBM-7094) con lo cual éstas eran mucho más comunes.



Ken Thompson de los Laboratorios Bell utilizó una minicomputadora en desuso, una PDP-7, para reescribir una versión reducida de MULTICS. Esta versión evolucionó a lo que hoy en día conocemos como UNIX.



Generación de las Computadoras Personales o Cuarta Generación (1980-2008)

Con el advenimiento de los microprocesadores, se abrió la puerta a que cada persona tuviera su propia computadora de uso personal (personal computer). Los microprocesadores siguen la ley empírica llamada ley de Moore, formulada por el mismísimo Gordon Moore, socio fundador de Intel. Que dice que la densidad de transistores por unidad de superficie se duplica cada 18 meses. Esto logró que en esta época el desarrollo de la capacidad de cómputo fuera acompañada con la necesidad de generar tiempos de respuesta muy reducidos y de alta calidad en lo que respecta a la interfaz gráfica de los sistemas.

inicialmente en esta Época dominaron el mercado MSDOS y UNIX. Ambas usaban la arquitectura intel x86. MSDOS de Microsoft evolucionó en el conocido sistema Windows 3.11 que posteriormete fuera integrado a DOS y se convirtiera en el sistema Operativo Windows 95. Por su lado de una forma un poco mas lenta pero constante UNIX que se habia convertido en un sistema propietarios empezo a tener clones no basados en su código fuente, entre ellos MINIX y Linux.



Linus Benedict Torvalds



Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torv...@kruuna.helsinki.fi)

PS. Yes - it's free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-).

Por otro lado cuando AT&T retiró la licencia de uso de UNIX a la universidad de Berkely, la misma promovió la construcción de un sistema operativo basado en UNIX al cual le serian agregados los aportes hechos por esta universidad llamado BSD. De este sistema operativo deriban SunOS, FreeBSD, NetBSD, PC-BSD, OpenBSD y Mac OS X.

Generación de los SmartPhones o Quinta Generación (2008 - Actualidad)

A partir del desarrollo de la telefonía celular en los llamados smartphones, los cuales en la actualidad poseen hardware que nada tienen que envidiar a las computadoras de escritorio, que cabe en la palma de una mano. Los sistemas operativos han evolucionado para ser ejecutados en estos dispositivos. Estos sistemas operativos son llamados sistemas operativos móviles. Es el caso de Android, basado en linux y de iOS basado en Mach que a su vez se basa en FreeBSD.

Los Sistemas Operativos

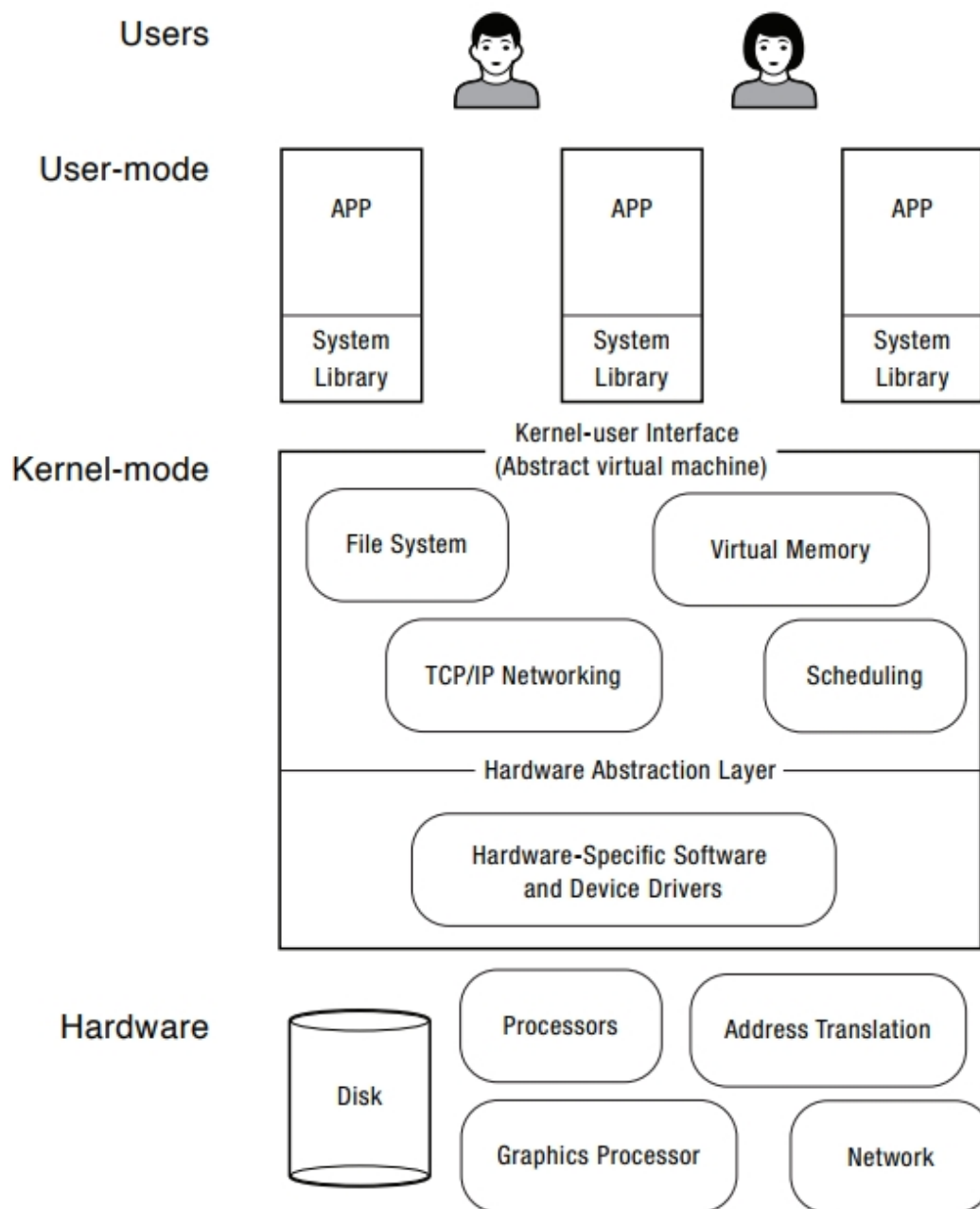
un sistema operativo es el software encargado de hacer que la ejecución de los programas parezca algo fácil. La forma principal para llegar a lograr esto es mediante el concepto de virtualización. esto es, el sistema operativo toma un recurso físico (la memoria, el procesador, un disco) y lo transforma en algo virtual más general, poderoso, fácil de usar.

Un sistema operativo deberá virtualizar varios recursos físicos:

- El procesador
- La memoria
- La persistencia (dispositivos de almacenamiento).

Por lo pronto se puede decir que un sistema operativo juega simultáneamente tres roles según varios autores DAH, ARP:

- **Referee:** Un OS gestiona recursos compartidos entre diferentes aplicaciones, que se encuentran ejecutándose en la misma máquina física. Por ejemplo:
 - un OS puede frenar la ejecución de una aplicación e iniciar la ejecución de otra.
 - los OS aíslan a cada aplicación de las demás que se encuentran corriendo en la misma computadora.
 - Por ende un OS tiene que protegerse a sí mismo y a las demás aplicaciones que se están ejecutando en la misma computadora.
 - Y dado que todas estas aplicaciones comparten los mismos recursos físicos el OS decide que aplicación usa un determinado recurso y cuando.
- **Ilusionista:** Un OS debe proveer una abstracción del hardware para simplificar el diseño de aplicaciones. Imaginen lo complejo que sería escribir el clásico hola mundo no se tiene que pensar en que lugar de la memoria física este se encuentra almacenado, ni como esta memoria se comparte con otros datos y aplicaciones. El sistema operativo provee la ilusión de que se dispone de toda la memoria para almacenar al programa, cuando realmente se sabe que la memoria ppal de la computadora es finita.
- **Pegamento:** Un OS debe proveer una serie de servicios comunes que faciliten un mecanismo que permita compartir, por ejemplo, información entre las aplicaciones.... "Cut & Paste" por ejemplo ... este mecanismo es uniforme en todo el sistema. Otro ejemplo es el "look and feel" de la interfaz de usuario. Tal vez uno de los más importantes sea el mecanismo de acceso a los dispositivos de entrada y salida del sistema, de forma tal que las aplicaciones puedan usarlos independientemente de la marca y modelo de los mismos.



- En el más bajo nivel se encuentra el hardware o recursos de la máquina:
 - CPU,
 - memoria principal,
 - dispositivos de red,
 - dispositivos gráficos,
 - dispositivos de almacenamiento, etc.

El hardware provee primitivas que el sistema operativo puede utilizar para aislar fallas y para sincronización.

- El Sistema Operativo se ejecuta como la capa de software de más bajo nivel en la computadora. Este contiene:
 - por un lado una capa para la gestión de dispositivos específico
 - y por otro una serie de servicios para la gestión de dispositivos agnosticos del hardware que son utilizados por las aplicaciones. Estas dos capas suelen ser conocidas como el kernel del sistema operativo. Cuando código fuente de esta capa es ejecutado

la computadora pasa a un estado llamado **Modo Supervisor**.

- Las aplicaciones se ejecutan en un **contexto aislado, protegido y restringido** y mediante el uso de funciones que se encuentran bibliotecas pueden utilizar los servicios de acceso al hardware o recursos que el kernel proporciona. El contexto de ejecución de las aplicaciones se denomina **User Mode** o modo usuario, mas restrictivo, aislado y controlado.

Por lo dicho anteriormente **existen dos modos de ejecución**

La perspectiva del usuario :

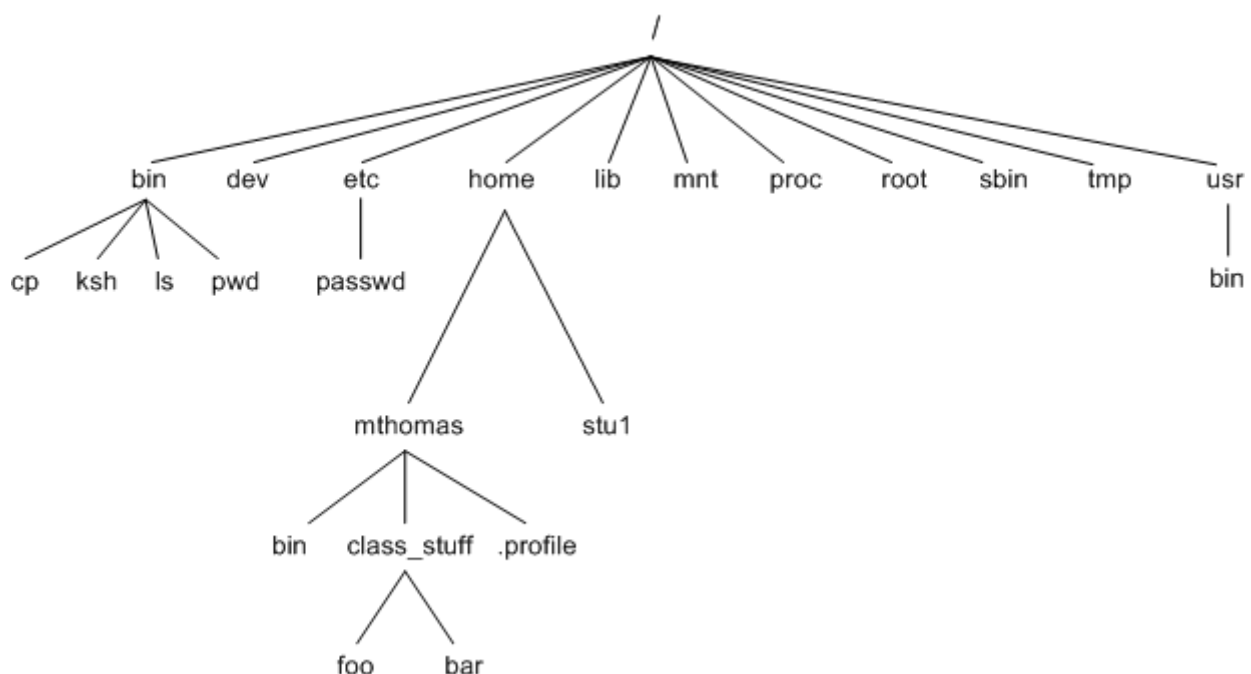
Desde el punto de vista del usuario lo que “se ve” es:

1. El **Sistema de Archivos** o File System.
2. El **Entorno de Procesamiento**.
3. Los **Bloques Primitivos de Construcción** .

El Sistema de Archivos

El sistema de archivos de Unix se caracteriza por:

- su **estructura jerarquica**
- **tratamiento consistente** archivos de datos
- la **habilidad de crear y borrar archivos**
- el **crecimiento dinámico** de los archivos
- la **proteccion de los archivos de datos**
- el **tratamiento de los dispositivos perifericos como si fueran archivos**.



- El Sistema de archivos de unix está organizado como un árbol con una única raíz (root) (que se escribe como "/"); cada nodo no hoja del sistema de archivo se denomina directorio de archivos, y las hojas del árbol pueden ser a su vez pueden ser archivos, directorios de archivos, archivos especiales, etc.
- El nombre de un archivo está dado por el path (camino) para localizarlo dentro de la estructura del sistema de archivos. "/etc/passwd"
- El sistema operativo no tiene noción de cómo es la estructura interna en el que se guardan.
- El sistema operativo Unix provee una lista de programas que vienen junto al sistema operativo, estos programas normalmente se denominan comandos, para realizar operaciones sobre los componentes del sistema de archivos:
 - **touch**: este comando permite crear un archivo o marcarlo como modificado.
 - **rm**: este comando permite borrar un archivo.
 - **mkdir**: este comando permite crear un directorio.
 - **rmdir**: este comando permite eliminar un directorio.
 - **ls**: este programa permite listar el contenido de un directorio.
 - **pwd**: este comando permite saber en que parte del sistema de archivos se está.
 - **cat**: este comando permite ver el contenido de un archivo.
 - **cp**: copia un archivo.
 - **mv**: cambia la locación de un archivo.

Existe una lista bastante extensa de comandos unix, aquí se deja una lista de los comandos que tienen que estar en una distribución de UNIX [Unix Commands](#)

El Entorno de Procesamiento

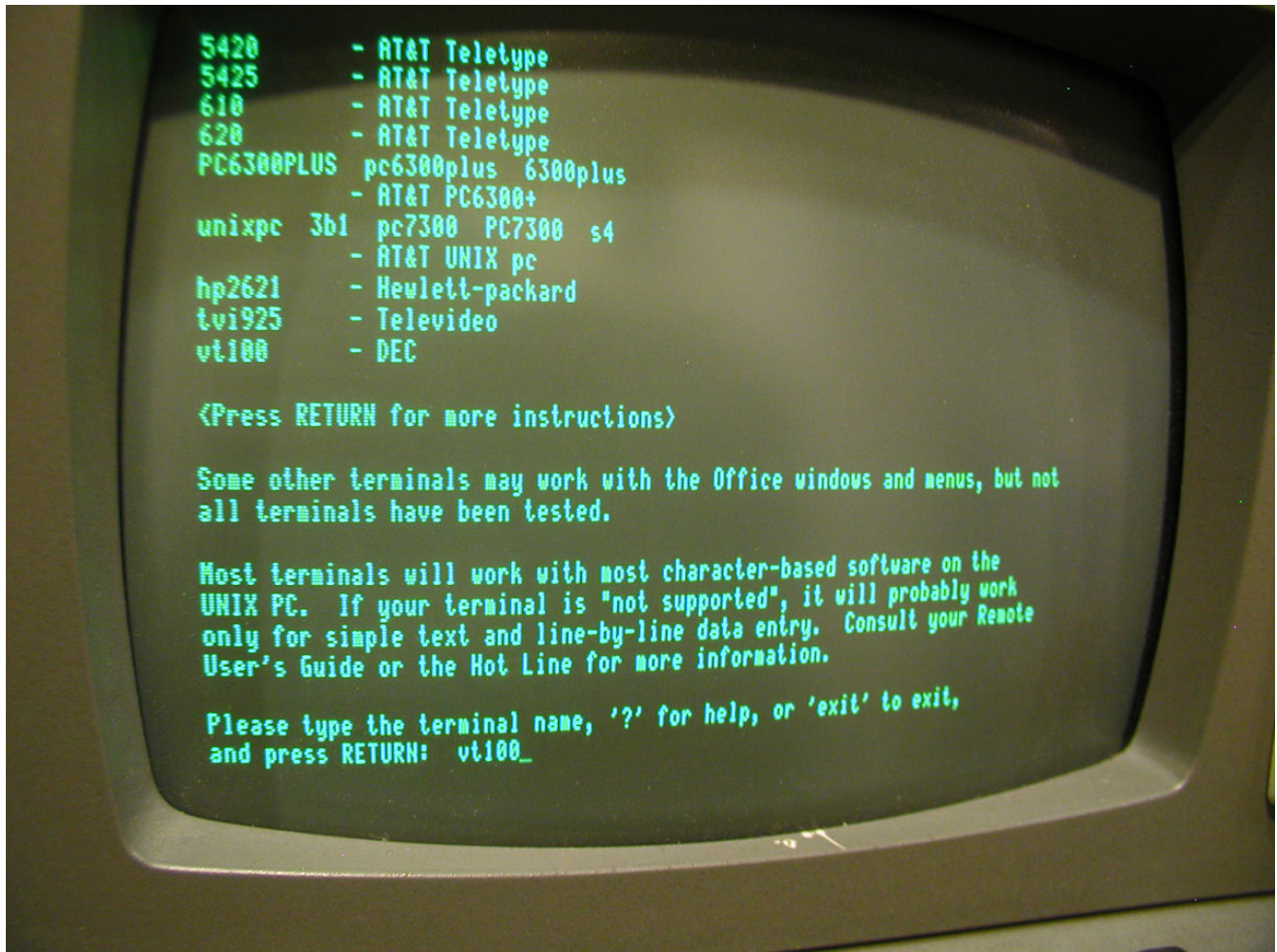
Un programa es un archivo ejecutable un proceso es una instancia de un programa en ejecución.

- Muchos procesos pueden estar ejecutándose a la vez, no existe un límite logico para la cantidad de estos.
- Existen varios comandos relacionados con los procesos:
 - **ps**: muestra los procesos que ha lanzado un determinado usuario.
 - **top**: muestra la lista de comandos que utilizan mas recursos.
 - **who**: muestra los usuarios conectados.
 - **kill**: mata a un proceso.
 - **free**: muestra la memoria libre del sistema.
- El interprete de comandos o *shell* es el primer programa que se ejecuta (en un unix) en modo usuario, este a su vez ejecuta el comando de login. El interprete de comandos captura las teclas que son presionadas en el teclado y las traduce a nombres de comandos.
- El shell puede interpretar tres tipos de comandos:

1. Comandos Ejecutables simples prorammas , por ejemplo el ls

2. **Shell scripts** que es un archivo ejecutable, consistente en líneas de comandos ejecutables

3. **Estructuras de control del Shell** *if-then-else-fi*



- El shell ejecuta los comandos buscando en ciertos directorios en una determinada secuencia.
- dado que el shell es un programa no forma parte del Kernel del sistema operativo
- El sistema operativo provee también unas System Calls para la creación y manipulación de procesos:
 - **fork()**: crea un proceso hijo a partir de un proceso padre.
 - **exec()**, **execve()**: creación de procesos.
 - **wait()**

Bloques Primitivos de Construcción

La idea de unix es la de proveer ciertos mecanismos para que los programadores construyan a partir de pequeños programas otros más complejos.

Esta idea está basada en la ya vieja y conocida estrategia de **Divide y Conquista**.

Estos mecanismos son usables desde el shell:

- El redireccionamiento de entrada y salida de datos. Los procesos acceden normalmente a tres archivos estandar: `_stdin_`, `_stdout_` y `_stderr_`
- El operador `>` redirecciona el standar output a un archivo. `ls >output`
- El operador `<` redirecciona el standar input tomando los datos de un archivo. `iconv -f ISO-8859-15 -t UTF-8 < Introduccion_a_la_programacion.tex > intro.tex`
- Las tuberías o los pipes. Permite que un stream de datos sean pasados entre dos procesos uno escritor y el otro lector. Del extremo izquierdo del caño se redirecciona el standar output de ese proceso al extremo derecho del caño el cual redirecciona el standar input del otro proceso.