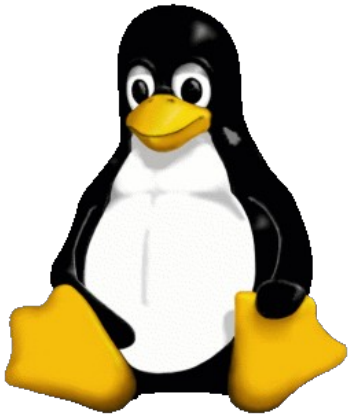
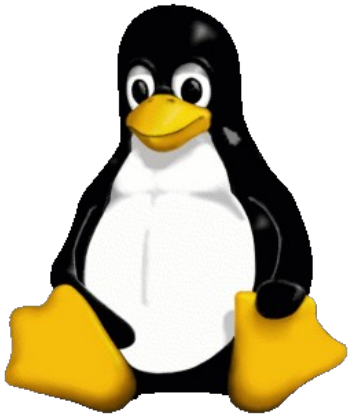


Introducción a UNIX®
Unidad 1

Daniel Millán, Nicolás Muzi, Eduardo Rodriguez

Marzo 2024, San Rafael, Argentina

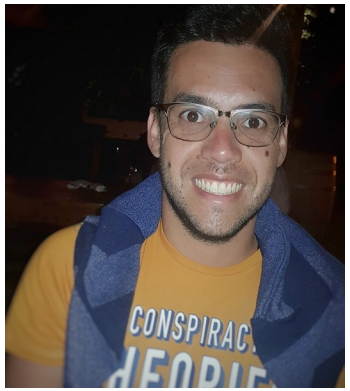
Curso basado en uno propuesto por ***William Knottenbelt***, UK, 2001



Introducción a UNIX®

Daniel Millán

Nicolas Muzi



Eduardo Rodriguez

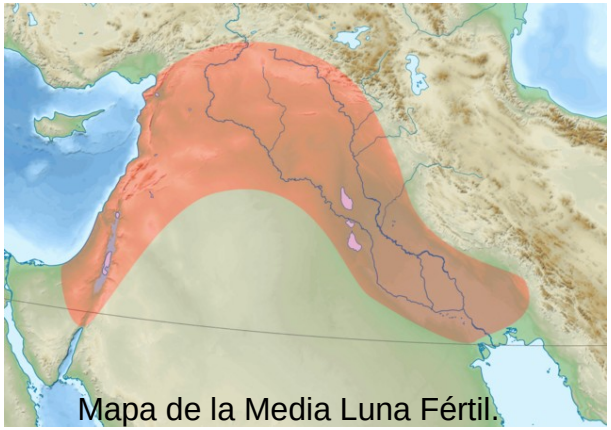
Más información sobre nuestras actividades en:

https://sites.google.com/fcai.uncu.edu.ar/mocca
i

https://icai.conicet.gov.ar/

Reseña histórica

En la antigua Mesopotamia, en la civilización Sumeria, tuvo su origen el **sistema sexagesimal,** es un sistema de numeración posicional que emplea como base aritmética el número **60.**



Mapa de la Media Luna Fértil.

El sistema sexagesimal se usa para medir tiempos (horas, minutos y segundos) y ángulos (grados) principalmente.

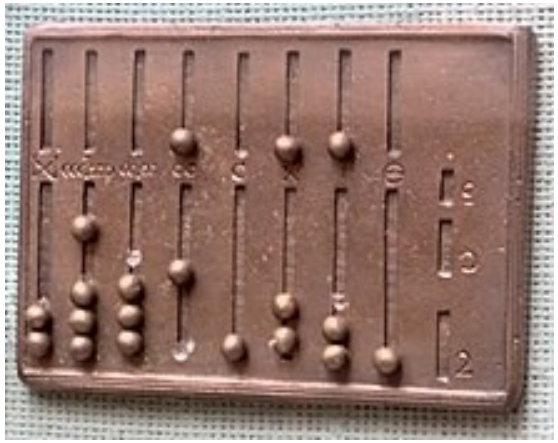
En Babilonia se dividió la circunferencia en 360 arcos iguales. Cada una de esas partes recibió el nombre de grado y a cada una de ellas se le asignó un dios.

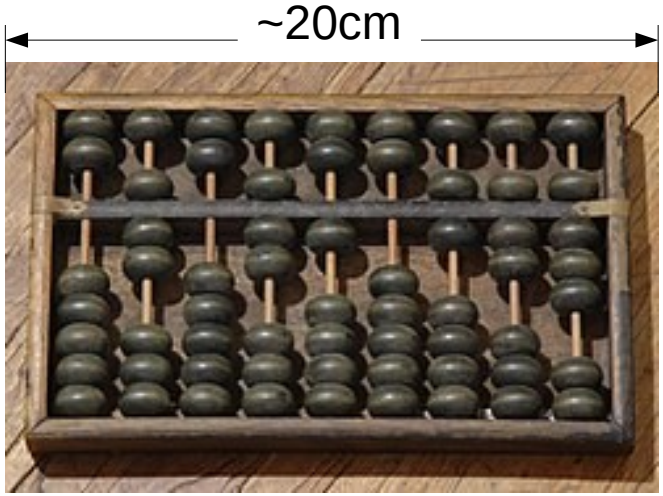
Zigurat de la ciudad de Ur que se ha conservado hasta nuestros días. Las ciudades sumerias se erigían alrededor de estos y en ellos los patesi realizaban ritos sagrados. (Wiki)

Reseña histórica

Ábaco: permite realizar operaciones aritméticas sencillas, su origen se remonta a la antigua Mesopotamia, más de 2000 años antes de nuestra era.

Wikipedia

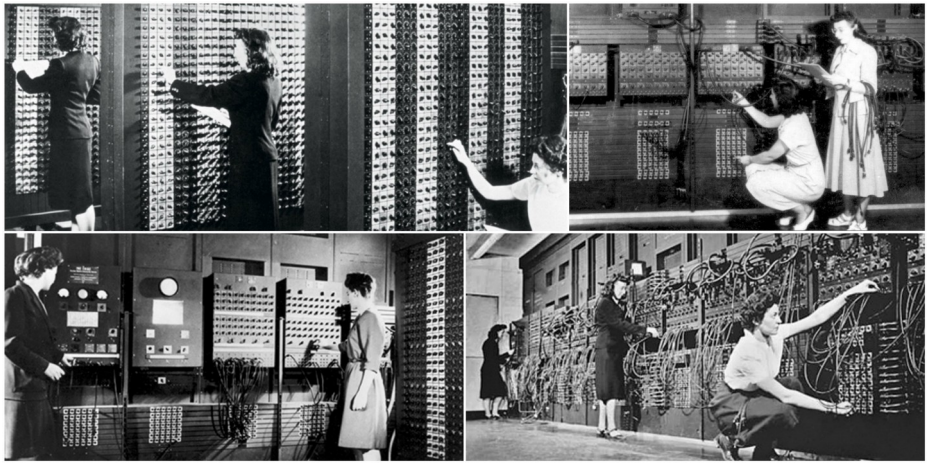




≤200 a.C.: Ábaco Romano.

≤200 a.C.: Ábaco Chino. El **suanpan** es un ábaco de origen chino.

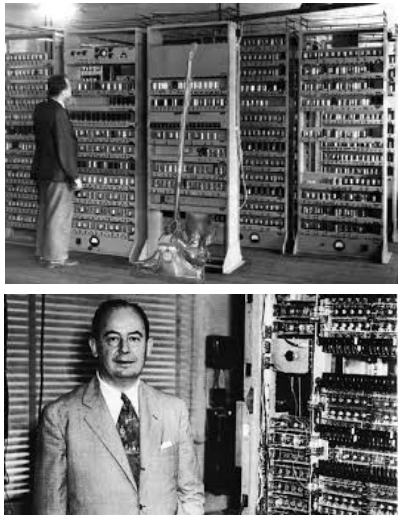
Reseña histórica



1946: La **ENIAC** fue inicialmente diseñada para calcular tablas de tiro de artillería destinadas al Laboratorio de Investigación Balística de la Universidad de Pensilvania, para el ARMY USA.

1645: Blaise Pascal inventa la **pascalina,** una de las primeras calculadoras mecánicas. Funcionaba a base de ruedas de diez dientes, cada uno representaba un dígito del 0 al 9.

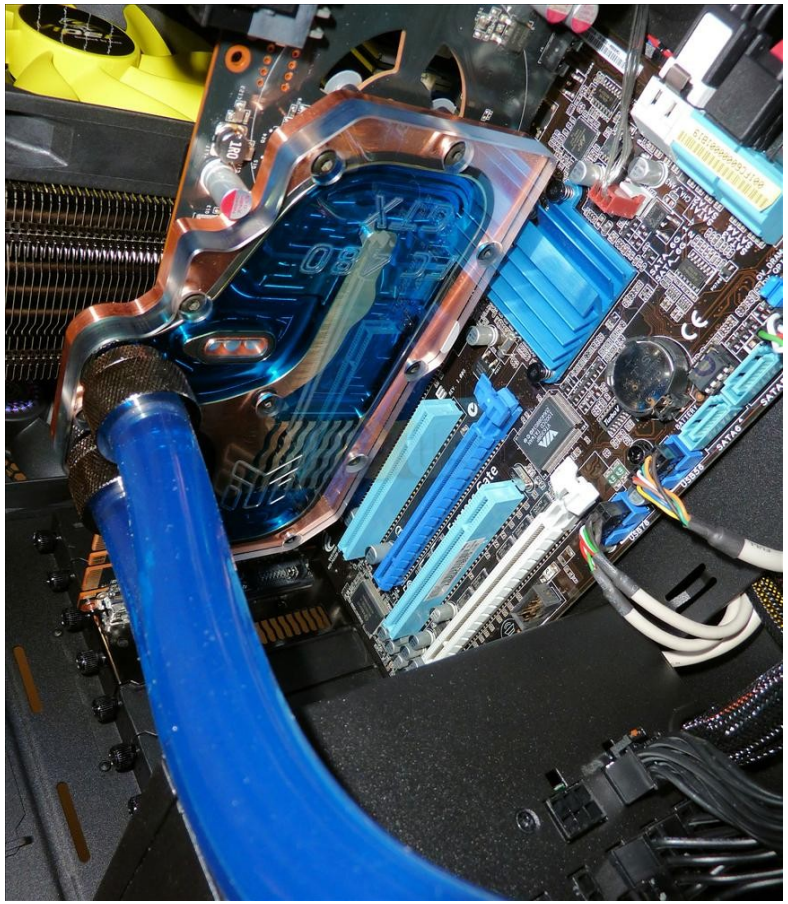
1949: La **EDVAC** fue la primer computadora de programas almacenados electrónicamente en forma binaria.
Wikipedia.



John von Neumann

2010: Scan 3XS Cyclone PC

- primer tarjeta gráfica de NVIDIA con refigeración líquida
- overclocked GeForce GTX 480, opera a 852MHz (701MHz).
- procesador *i7 920*, *overclocked* a 4GHz.
- £1,646.84, incluyendo impuestos.



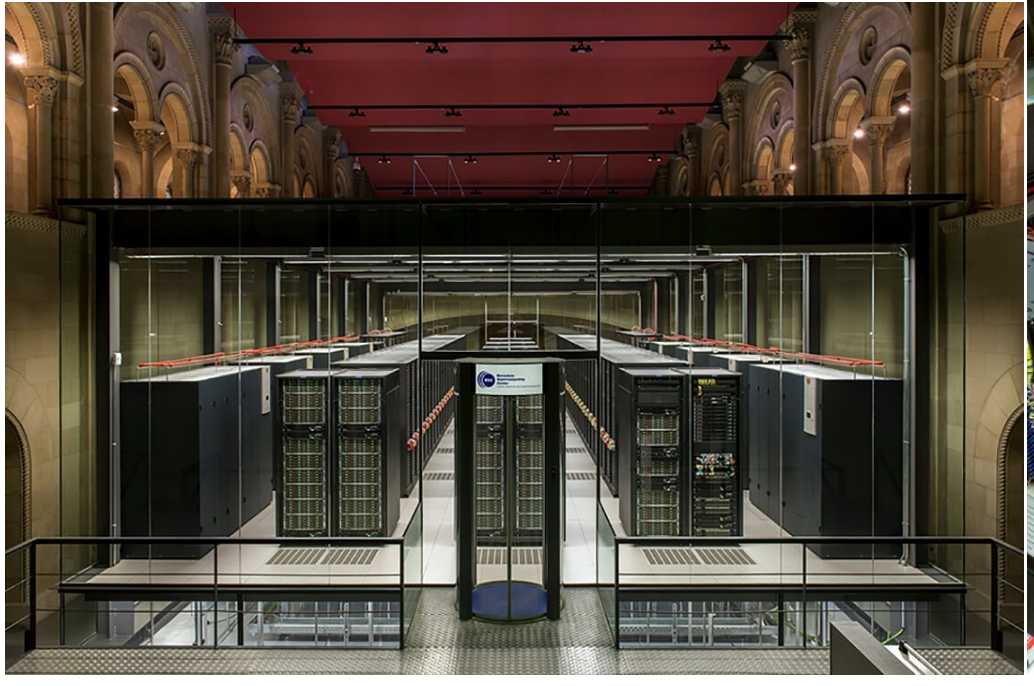
http://hexus.net/tech/reviews/systems

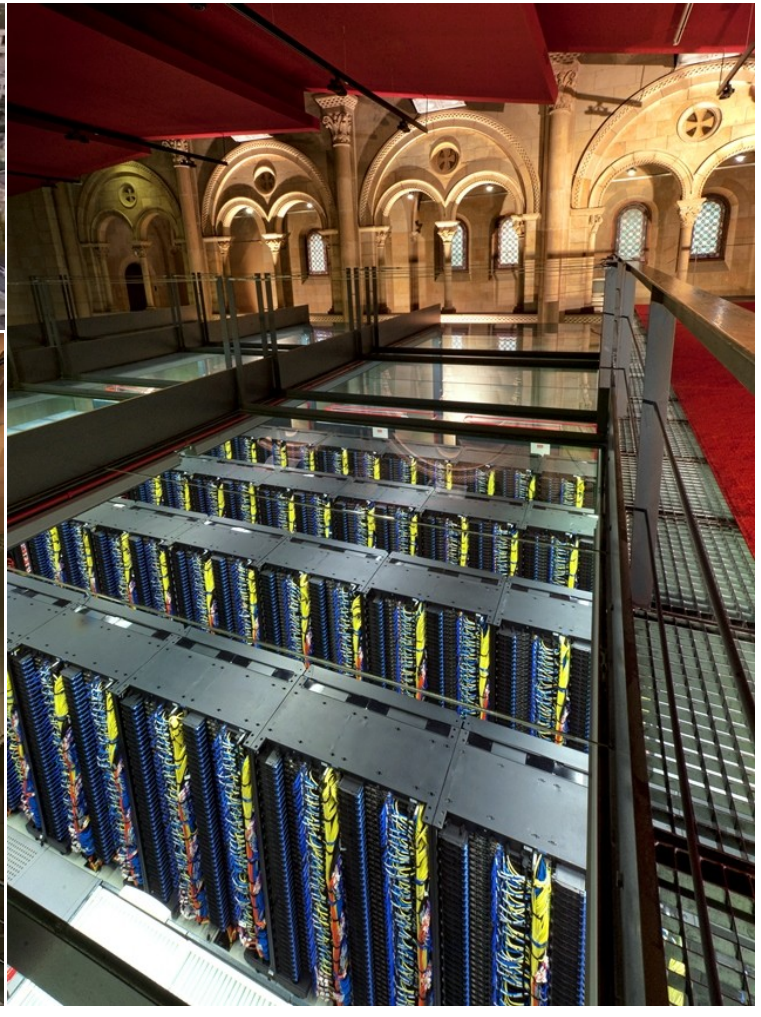
https://www.bsc.es/

2019: MareNostrum Barcelona

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

- *MareNostrum* es el supercomputador más potente de España, el quinto más rápido de Europa y el 25° del mundo (nov – 2018).





2019: MareNostrum Barcelona

Áreas de investigación

- **Composición atmosférica**: calidad del aire, aerosoles y como estos dispersan y absorben la radiación solar, ciudades inteligentes y la optimización del transporte y la salud humana.
- **Big Data**: herramientas visuales y algorítmicas para analizar y estudiar grandes volúmenes de datos.
- **Bioinformática**: integración, almacenamiento y transmisión de gran volumen de datos clínicos y datos de simulaciones, diseño de fármacos.
- **Biomecánica:** sistema cardiovascular y sistema respiratorio.
- **Predicción climática:** gestión de la agricultura y del agua, el pronóstico oceánico, estudio de los ciclones tropicales, estudio de dónde es más eficiente instalar un molino de viento.
- **Computación en la nube:** informática energética y optimización de los centros de datos.

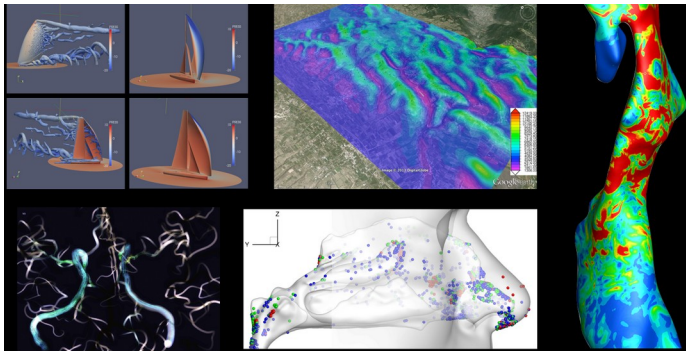
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

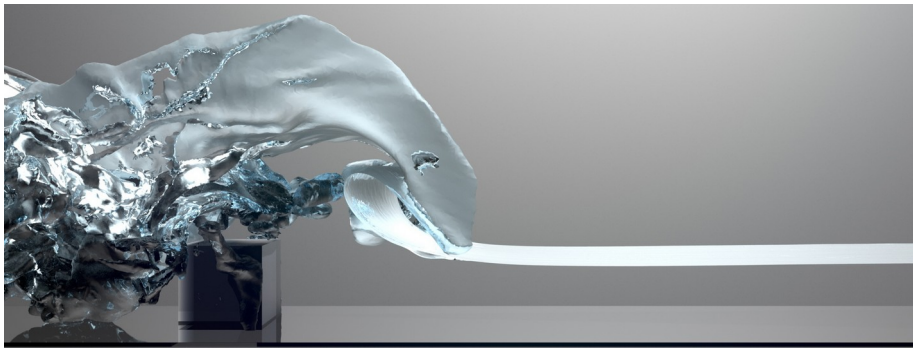
https://www.bsc.es/

2019: MareNostrum Barcelona

Áreas de investigación

- **Simulación de ingeniería:** reducción de las emisiones contaminantes, computación en mecánica de fluidos, mecánica nolineal de sólidos.





- **Geofísica:** terremotos, detección de la presencia de fluidos a grandes profundidades bajo la superficie de la Tierra, propiedades de la superficie de la Tierra.
- **Simulación social:** evolución cultural, eficiencia energética, seguridad pública de cara a tener ciudades inteligentes y resistentes.

2019: MareNostrum Barcelona

Alya Red

Proyecto Alya Red, su video promocional fue elegido mejor vídeo científico del 2012 por la National Science Foundation norteamericana y la revista Science. En él se explica cómo se crean los modelos con los que se simula el funcionamiento de un corazón, intentando imitar el comportamiento de los diferentes tejidos y de cómo las señales eléctricas viajan por su interior.

La tarea es tan compleja que para poder analizarlo con precisión se emplea el ordenador Mare Nostrum del Centro de Supercomputación de Barcelona.

Unidad 1: **Sistema Operativo Unix**

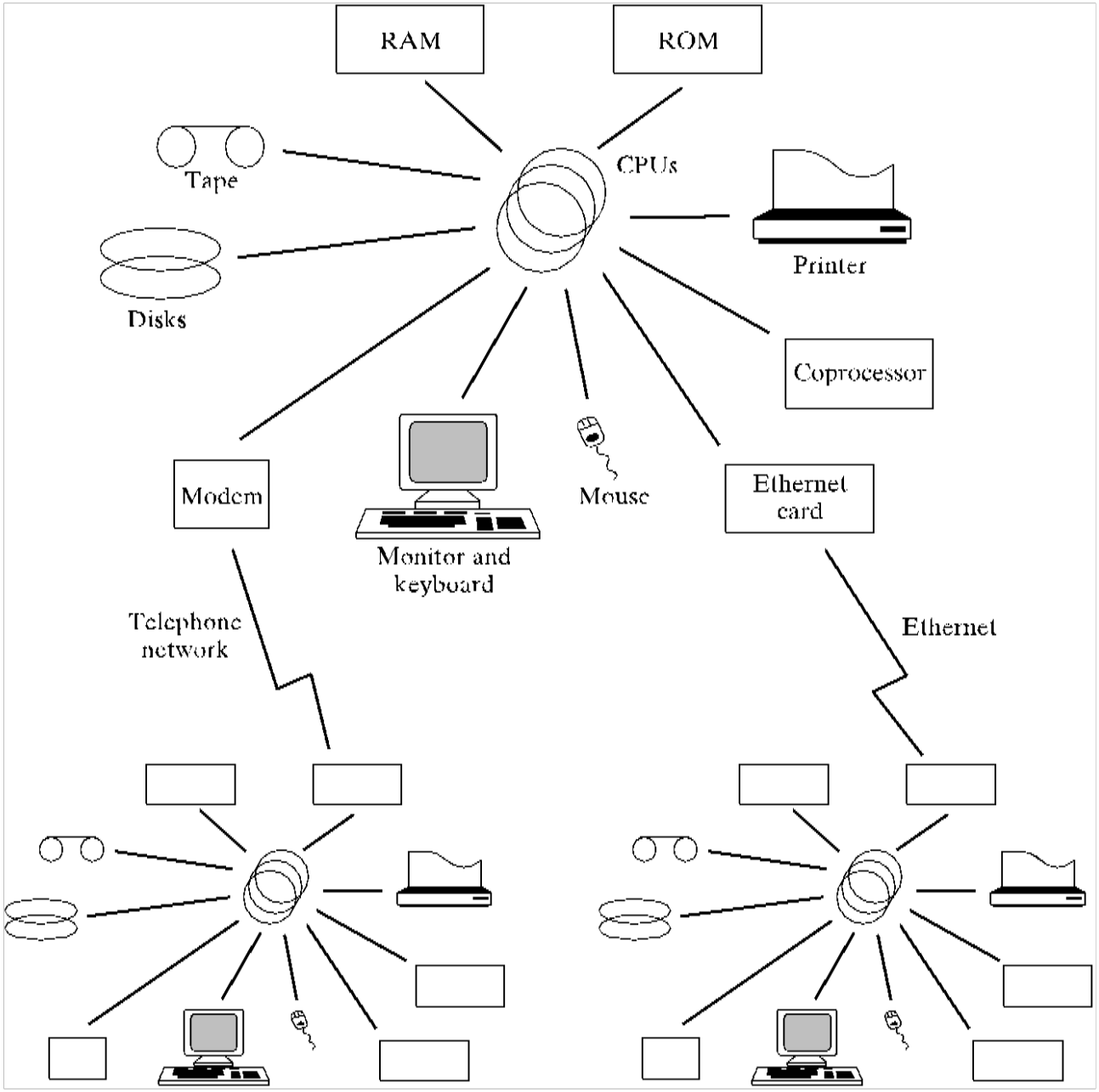
1. ¿Qué es un sistema operativo?
2. Breve introducción a la historia de Unix.
3. Arquitectura del sistema operativo Linux.
4. Inicio/Salida de sesión en sistemas Unix.
5. Cambio de contraseña.
6. Formato general de los comandos de Unix.
7. El sistema de ficheros Unix.
8. Típica estructura de directorios Unix.
9. Manejo de archivos y directorios.
10. Enlaces a ficheros (directos/simbólicos).
11. Especificación de múltiples nombres de archivo.
12. Comillas y caracteres especiales.


WELCOME, NEW USER

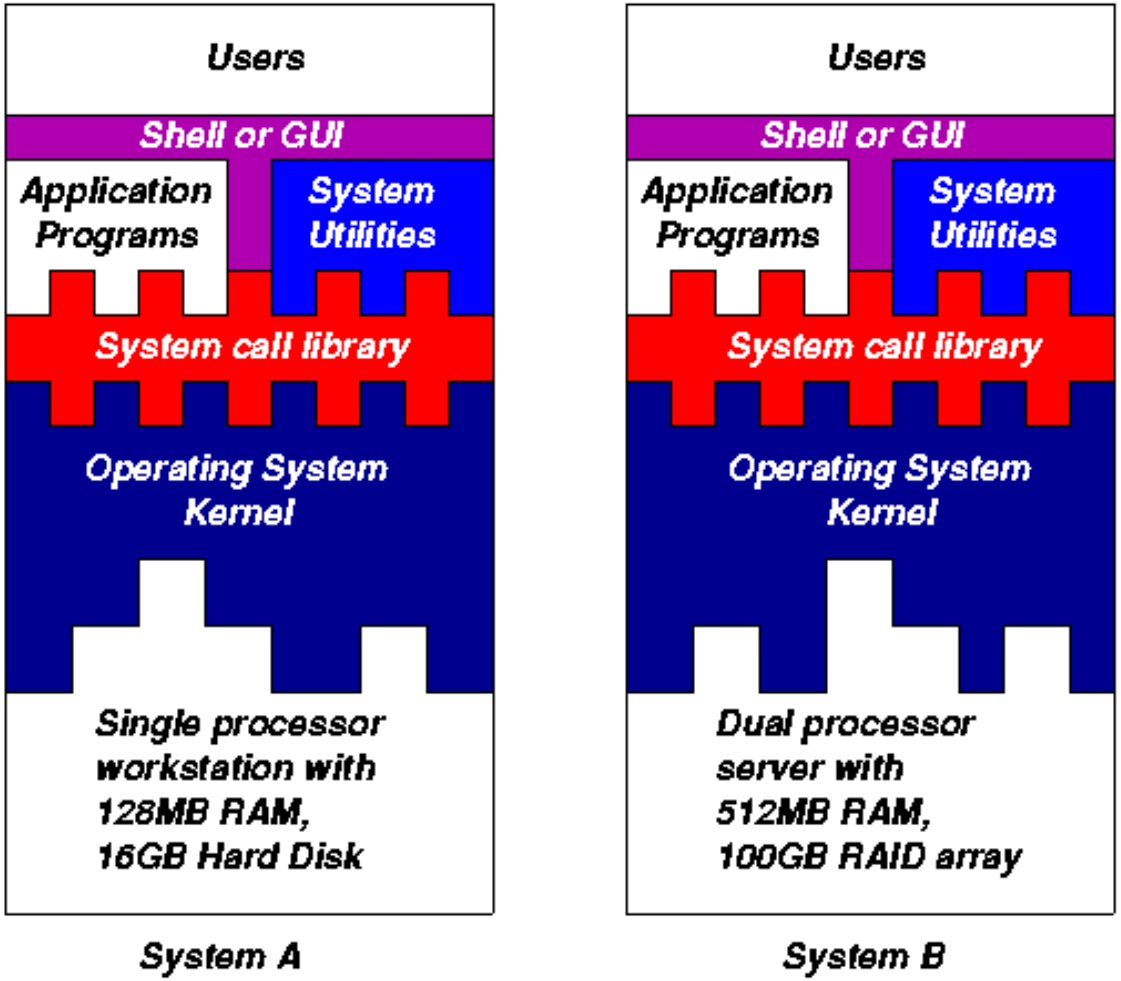
1. ¿Qué es un OS?

- Un sistema operativo (OS) es un gestor (administrador) de recursos.
- Se presenta en forma de un conjunto de rutinas de software que permiten a los usuarios y a los programas acceder a los recursos del sistema de una manera **segura, eficiente** y **abstracta.**
  - CPU, tarjetas de red, discos de memoria, módems, impresoras, etc…
  - **CPU**: *central processing unit*
    Unidad de Procesamiento Central.
  - El OS asegura un acceso seguro p.ej. Impresora.
  - El OS fomenta el uso eficiente de la CPU mediante suspensión de operaciones de *Entrada/Salida*.
  - El OS proporciona abstracciones tales como archivos en lugar de posiciones de memoria en discos (detalles de hardware están ocultos).

Wikipedia


RAM
ROM
Tape
CPUs
Printer
Disks
Coprocessor
Modem
Monitor and keyboard
Mouse
Ethernet card
Telephone network
Ethernet

1. ¿Qué es un OS?



Arquitectura genérica del sistema operativo

1. ¿Qué es un OS?

- El **núcleo del** OS controla de forma directa el hardware subyacente.
- El núcleo maneja dispositivos de bajo nivel, la memoria y la gestión del procesador.
- Servicios básicos del núcleo están disponibles para programas de nivel superior a través de una biblioteca de **llamadas al sistema**.
- **Los programas informáticos o aplicaciones** (procesadores de texto, hojas de cálculo, **Octave**) y **programas de utilidades del sistema** (buscador) hacen uso de las llamadas al sistema.
- Aplicaciones y utilidades del sistema se ponen en marcha mediante un **shell** (interfaz de órdenes de texto) o una **interfaz gráfica de usuario** que proporciona una interacción directa (mouse).

2. Breve historia de Unix

- UNIX ha sido un OS popular durante más de 4 décadas debido a que brinda un entorno:
  - **Multi-usuario**
  - **Multitarea**
  - **Estabilidad**
  - **Portabilidad**
  - **Altas prestaciones para trabajo en red**

“UNIX es un sistema operativo de propósito general, multiusuario, e interactivo para las computadoras PDP-11/40 y 11/45 de la Coorporación Digital Equipment.”

Unix documentation, 1974

2. Breve historia de Unix

Principios de diseño

- UNIX fue diseñado para ser un SO interactivo, multiusuario y multitarea:
  - **Interactivo** quiere decir que el sistema acepta órdenes, las ejecuta y se dispone a esperar otras nuevas.
  - **Multitarea** significa que puede realizar varios trabajos, denominados procesos, al mismo tiempo.
  - **Multiusuario** significa que más de una persona puede usar el sistema al mismo tiempo.
- UNIX fue diseñado por programadores para ser usado por programadores en un entorno en que los usuarios son relativamente expertos y participan en el desarrollo de proyectos de software.

UNIX → *No user friendly*

2. Breve historia de Unix

American Telephone and Telegraph
(Direct TV U$S 48G)

- **1960**: General Electric + MIT + Bell Labs (AT&T) desarrollan MULTICS
  - SO multi-usuario y multitarea en ordenadores centrales (cajas grandes)
  - MULTICS: ***MULT****iplexed* ***I****nformation and* ***C****omputing* ***S****ystem*

- **1969**: **Ken Thompson** (Bell Labs)
  - Crea un SO basado en MULTICS pero más sencillo en una **PDP-7** (mini PC 1965)
  - UNICS: ***UN****iplexed* ***I****nformation and* ***C****omputing* ***S****ystem*
  - Poca memoria y potencia llevan a utilizar comandos cortos: **ls, cp, mv**…
  - El lenguaje de programación en que fue escrito UNICS se llamaba B
- **1971**: Se une **Dennis Ritchie**
  - Crea el primer compilador de C y se reescribe el núcleo de **UNIX en C** (1973)
  - Mejora de la portabilidad
  - Se lanza la quinta versión de UNIX a las Universidades en 1974 (GRATIS)
- **1978**: Se separan dos grandes ramas: SYSV (AT&T y otras empresas) y BSD (Berkeley Software Distribution de la UCB) ➔Incompatibles!

In 1975, Ken Thompson took a sabbatical from Bell Labs and came to Berkeley as a visiting professor, he helped to install Version 6 Unix .

2. Breve historia de Unix

Un **PDP-7** modificado, en restauración en Oslo, Noruega. **Wiki**

Ken Thompson y Dennis Ritchie.
https://www.timetoast.com/timelines/history-of-unix

2. Breve historia de Unix

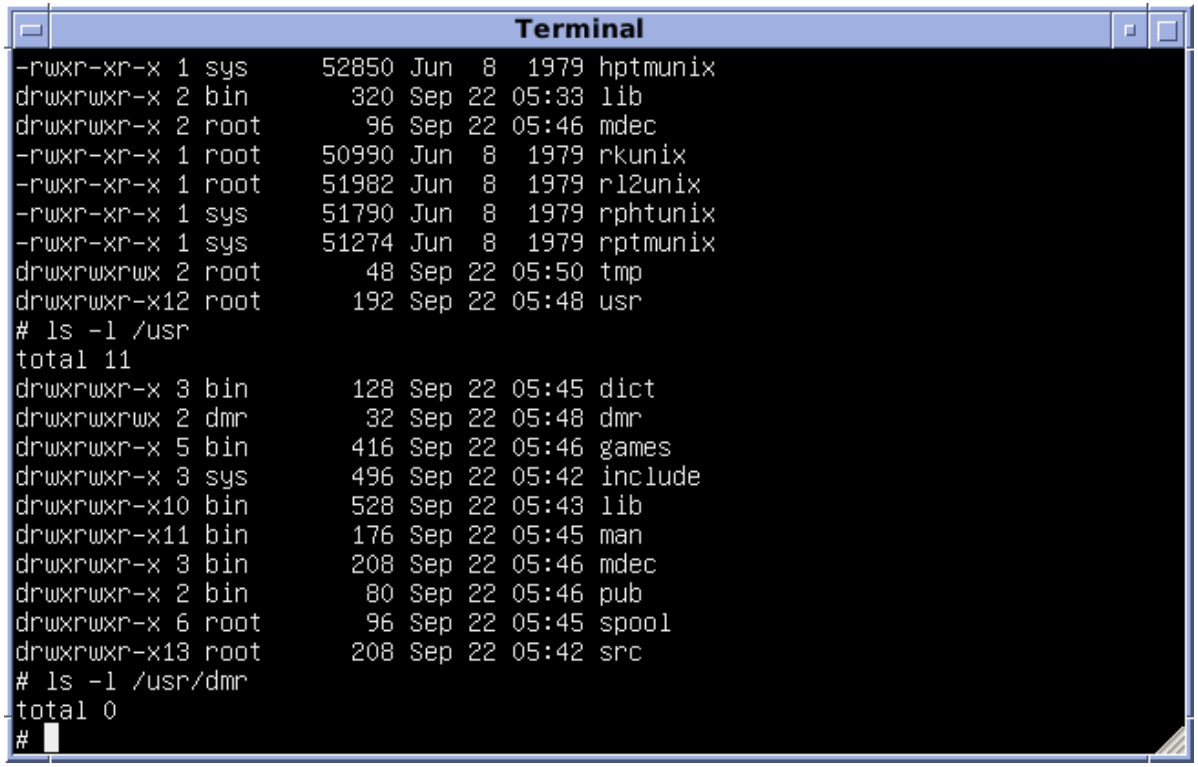
Feature	Typical SYSV	Typical BSD
kernel name	/unix	/vmunix
boot init	/etc/rc.d directories	/etc/rc.* files
mounted FS	/etc/mnttab	/etc/mtab
default shell	sh, ksh	csh, tsch
FS block size	512 bytes->2K	4K->8K
print subsystem	lp, lpstat, cancel	lpr, lpq, lprm
echo command (no new line)	echo “\c”	echo -n
ps command	ps -fae	ps -aux
multiple wait syscalls	poll	select
memory access syscalls	memset, memcpy	bzero, bcopy

Algunas diferencias entre SYSV y BSD se encuentran en:

- la estructura del sistema de archivos,
- los nombres y las opciones de utilidades del sistema,
- la bibliotecas de llamada del sistema,
- etc.

2. Breve historia de Unix

- **1979**: Aparece la Séptima Edición Unix, (Versión 7 o simplemente V7), fue una importante versión del sistema operativo Unix actual.



Los laboratorios Bell liberan una última distribución antes de la comercialización de Unix por AT&T. Muchas de sus características siguen hasta el presente.

En 1991 Linus Torvalds invierte U$S3500 (en cuotas) en un ordenador con el microprocesador 80386 de Intel, de 33 MHz y 4 MB de memoria RAM, dado que no estaba conforme con el OS Mimix decide crear el **OS Linux** para conectarse a su Universidad.

https://www.lleox.org/2016/te-lo-resumo-asi-nomas-linux-la-historia/

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)

Newsgroups: comp.os.minix

Subject: What would you like to see most in minix?

Summary: small poll for my new operating system

Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>

Date: **25 Aug 91 20:57:08 GMT**

Organization: University of Helsinki

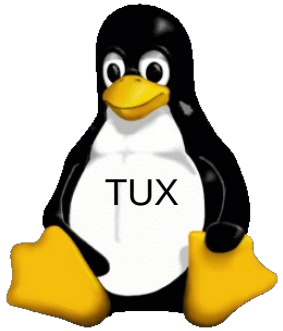
Hello everybody out there using minix –

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 (*torvalds@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 :-(.

2. Breve historia de Unix

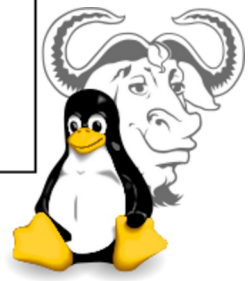


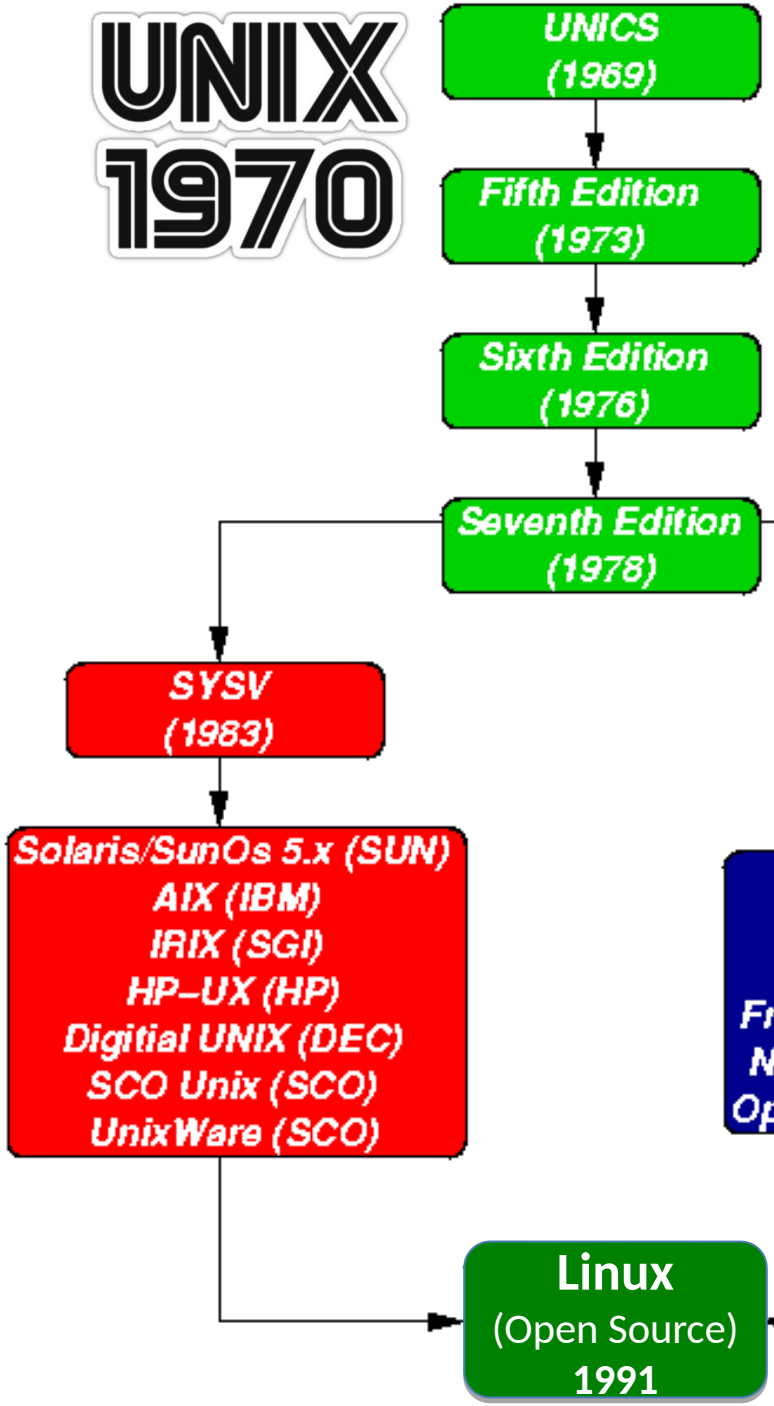
- **1991**: Linus Torvalds, un estudiante finlandés de Ciencias de la Computación diseña Linux un código abierto del SO UNIX para PC
  - No es SYSV ni BSD, pero incorpora características de cada uno (p.ej. al estilo SYSV archivos de inicio, pero con una disposición del sistema de archivos del tipo BSD).
  - Cumple con un conjunto de estándares de IEEE (*Institute of Electrical and Electronics Engineers*) llamado POSIX (*Portable Operating System Interface*)
  - Para maximizar la portabilidad del código, Linux *típicamente* soporta SYSV, BSD y llamadas al sistema de POSIX.
  - Linux ha generado que miles de personas colaboren voluntariamente durante >25 años mejorando el núcleo y programas de aplicación.
  - Diferentes distribuciones: Debian, Suse, RedHat, Ubuntu, etc
  - Portable a diferentes arquitecturas de procesadores como Intel, AMD, SPARC…
  - *Fácil* de usar e instalar y viene con un conjunto completo de utilidades y aplicaciones, incluyendo el sistema de gráficos X, entornos GNOME y KDE GUI.

In **1996** Larry Ewing released Tux under the following condition:
Permission to use and/or modify this image is granted provided you acknowledge me lewing@isc.tamu.edu and The GIMP if someone asks.
TUX: (T)orvalds (U)ni(X) or tuxedo abbreviation... how knows?

UNIX 1970


LIVE FREE OR DIE
UNIX*
TRADEMARK OF BELL LABS*




UNICS
(1969)
Fifth Edition
(1973)
Sixth Edition
(1976)
Seventh Edition
(1978)
SYSV
(1983)
BSD
(1979)
Solaris/SunOs 5.x (SUN)
AIX (IBM)
IRIX (SGI)
HP-UX (HP)
Digitial UNIX (DEC)
SCO Unix (SCO)
UnixWare (SCO)
SunOs 4.x (SUN)
ULTRIX (DEC)
NextStep (NeXT)
FreeBSD (Open Source)
NetBSD (Open Source)
OpenBSD (Open Source)
Linux
(Open Source)
1991

3. Arquitectura del SO Linux

Linux tiene todos los componentes de un SO tipo UNIX:

- Núcleo: facilita acceso seguro a distintos programas al hardware (tarjetas gráficas y red, discos duros, etc), decide qué programas utilizan hardware y cuánto tiempo (multiplexado), BSD/SYSV llamadas de sistema, etc.
- Shells y GUIs:
  - Intérpretes de línea de comandos (shells) como en UNIX: **sh**: shell Bourne, **bash:** *Bourne again shell* y **csh**: *C shell*
  - Interface Gráfica (GUI, *Graphic User Interface*), gestores KDE y GNOME
- Utilidades del sistema: Herramientas poderosas que hacen una sola tarea extremadamente bien.
  - **cp** copia, **grep**: busca expresiones regulares (caracteres), **awk**: procesa datos definidos en archivos de texto, **sed**: editor de flujo de texto, demonios, etc.
- Programas de aplicación:
  - **emacs**: editor de texto, **gcc/g++**: compilador de C/C++, **latex**: lenguaje de composición de texto, **xfig**: paquete de dibujo vectorial, **StarOffice**, etc.

4. Inicio/Salida de sesión SO Unix

- **(TTY) terminales de texto:** Cuando se conecta a un ordenador UNIX de forma remota o al iniciar una sesión localmente usando una terminal de sólo texto, verá el símbolo:

  login: *pepe* (inicio de sesión)

- En este indicador, escriba en su username y presione el **Enter**. Recuerde que UNIX es sensible a mayúsculas (*pepe*, *Pepe* y *PEPE* son inicios de sesión diferentes).

  login: *pepe*
  password: ****** (contraseña)

- Escriba su contraseña y presione el **Enter**. Tenga en cuenta que la contraseña no se mostrará en la pantalla al escribir.
- Si escribe mal su nombre de usuario o la contraseña recibirá un mensaje y se mostrará nuevamente **login**:
- De lo contrario, el intérprete de comandos muestra algo como esto:

  $ | (el cursor parpadea ➔ esperando instrucción)

- Para salir de la sesión, escriba **exit** en el intérprete de comandos (si eso no funciona intente **logout**, y si eso no funciona pulse Ctrl-D).

4. Inicio/Salida de sesión SO Unix

Terminal Gráfica

- Si usted está entrando en un ordenador UNIX a nivel local, o si está utilizando un inicio de sesión remoto que soporta gráficos, observará los campos de usuario y contraseña.
- Introduzca su nombre de usuario y la contraseña de la misma manera que antes (*Nota*: tecla TAB permite moverse entre los campos).
- Una vez que se ha identificado, se le presentará un gestor de ventanas que se ve similar a la interfaz de Microsoft Windows. Para que aparezca una ventana de intérprete de comandos busque los menús o íconos que mencionan las palabras **shell**, **xterm**, **terminal emulator**, or **console**.
- Para cerrar la sesión de un gestor de ventanas gráficas, buscar opciones de menú similares a **Exit** o **Log Out**.

5. Cambio de contraseña

- Una de las cosas que debe hacer cuando se conecta por primera vez en un SO tipo UNIX es cambiar su contraseña. El comando de UNIX para cambiar su contraseña es passwd:

  $ passwd

- El sistema le pedirá la contraseña anterior, a continuación, introduzca su nueva contraseña (por duplicado).
- Recuerde los siguientes puntos al elegir su contraseña:
  - Evitar caracteres que pudieran no aparecer en todos los teclados, por ejemplo: '£', '€', 'ñ', 'ç', 'á', etc.
  - El eslabón más débil en la seguridad informática suelen ser las contraseñas de los usuarios. No la facilite a nadie. Evite las palabras del diccionario o palabras relacionadas con sus datos personales (p.ej., nombre novia/novio o el nombre de su nombre de usuario).
  - La contraseña de poseer al menos 7 u 8 caracteres de longitud y tratar de utilizar una combinación de letras, números y puntuación.

6. Formato de órdenes de Unix

- Una línea de comandos UNIX consiste en el nombre de un comando UNIX (en realidad el "comando" es el nombre de un programa de la **shell**, una utilidad del sistema o un programa de aplicación) seguido de sus "argumentos" (opciones y los nombres de archivo de destino).
- La sintaxis general para un comando UNIX es:

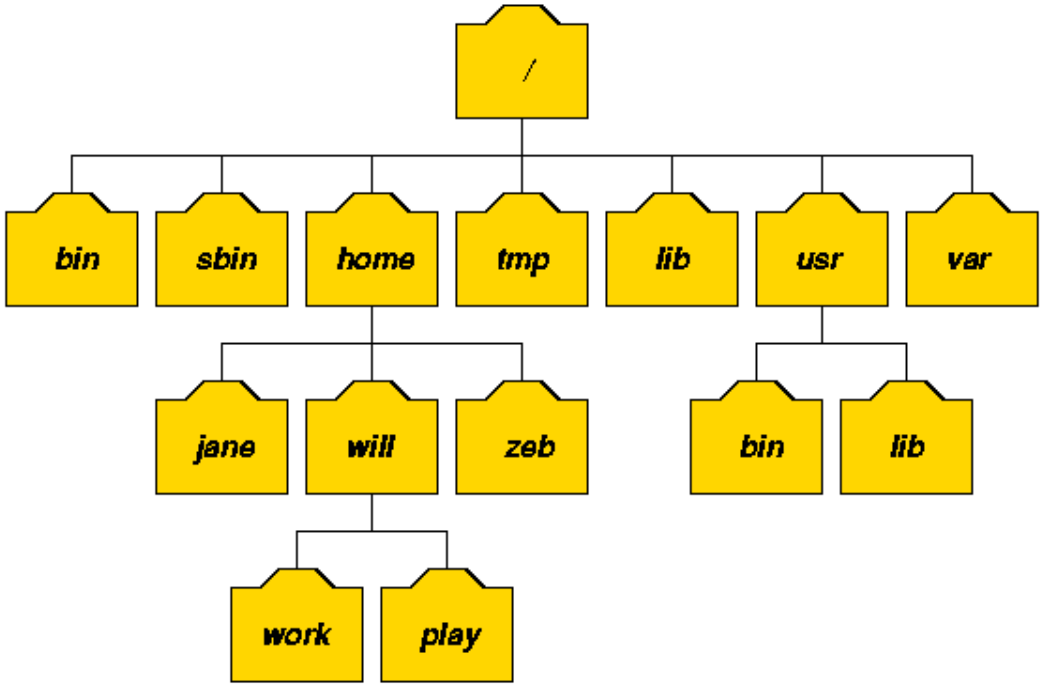
  $ **orden –opciones objeto**
- Aquí **orden** puede ser entendida como un verbo, **opciones** como un adverbio y **objeto**, como los objetos directos del verbo.
- En el caso de que querer especificar varias opciones, éstas no siempre tienen que ser listadas por separado (las opciones comúnmente se pueden concatenar después de un único guión).

7. El sistema de ficheros Unix

- El sistema operativo UNIX se basa en el concepto de un sistema de archivos que se utiliza para almacenar toda la información que constituye el estado a largo plazo del sistema.
- Cada elemento almacenado en un sistema de archivos de UNIX pertenece a uno de cuatro tipos:
  - **Archivos ordinarios**: contienen datos, texto, o información de programas. Para definirlos no utilizar *, **?**, **#** ni espacios (utilizar el guión bajo).
  - **Directorios**: carpetas que contienen archivos y otros directorios.
  - **Dispositivos:** Para proporcionar que las aplicaciones tengan fácil acceso a los dispositivos de hardware, UNIX permite que sean manejados mediante archivos.
  - **Enlaces**: es un puntero a otro archivo. Hay dos tipos de enlaces. **Enlace físico** es indistinguible de la del propio archivo. Un **enlace simbólico** es un puntero indirecto a un archivo, consiste en una archivo de directorio que contiene la dirección del archivo en el sistema de ficheros.

8. Estructura de directorios Unix

- El sistema de archivos de UNIX se presenta como una estructura jerárquica de árbol que está anclado en un directorio especial de alto nivel conocido como la raíz: **/**
- Debido a la estructura de árbol, un directorio puede tener muchos directorios secundarios, pero sólo un directorio padre.



8. Estructura de directorios Unix

- **/** El directorio "raíz".
- **/bin** Utilidades del sistema de bajo nivel esenciales.
- **/usr/bin** Utilidades del sistema de nivel superior y los programas de aplicación.
- **/sbin** Utilidades del sistema superusuario (para realizar tareas de administración del sistema).
- **/lib** Bibliotecas de programas (llamadas al sistema que se pueden incluir en los programas por un compilador) para las utilidades del sistema de bajo nivel.
- **/usr/lib** Bibliotecas de programas para programas de usuario de alto nivel.
- **/tmp** Espacio de almacenamiento de archivos temporales (se puede utilizar por cualquier usuario).
- **/home** Directorios de los usuarios que contienen espacio de archivos personales de cada usuario. Cada directorio es el nombre de sesión del usuario.
- **/etc** Archivos UNIX de configuración y la información del sistema.
- **/dev** Dispositivos de hardware.
- **/proc** Un pseudo sistema de archivos que se utiliza como una interfaz para el kernel. Incluye un subdirectorio para cada programa activo (o proceso).

9. Manejo de archivos y directorios

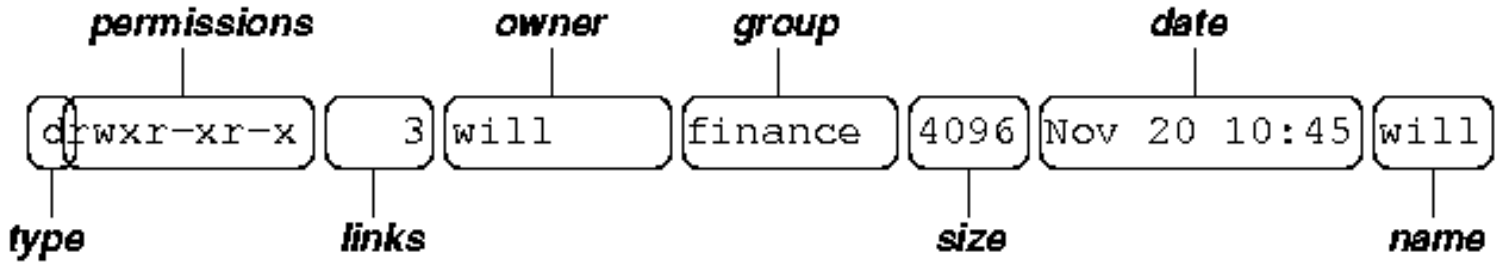
- Algunos de los comandos más importantes

pwd ruta absoluta a la ubicación actual en el sistema de archivos

ls muestra el contenido de un directorio (p/defecto el directorio de trabajo)

ls –a muestra todo incluso archivos ocultos (comienzan con .)

ls –a –l o equivalentemente **ls –al**



Tipo: **"d"** directorio, **"-"** archivo ordinario, **"l"** enlace simbólico

Permisos: tres derechos de acceso, lectura **"r"**, escritura **"w"** y ejecución **"x"**, y tres categorías de usuarios: propietario, grupo y otros (público en general).

Para conocer más opciones de los comandos puedes utilizar **man** o **info.**

man ls manual de usuario en línea del comando **"ls"** de UNIX (muy conciso = duro).

Nota: Si está instalado puede resultar más útil el comando **info** (no estándar).

9. Manejo de archivos y directorios

- Algunos de los comandos más importantes:

cd cambia el directorio de trabajo actual a un *path* que puede ser absoluto/relativo

$cd /home/pepe equivalente **$cd ~** equivalente **$cd**

mkdir crea un subdirectorio en el directorio de trabajo actual.

rmdir elimina un subdirectorio del directorio actual siempre que esté vacío.

cp se utiliza para hacer copias de archivos o directorios completos.

mv se utiliza para cambiar el nombre de los archivos / directorios y / o moverlos de un directorio a otro.

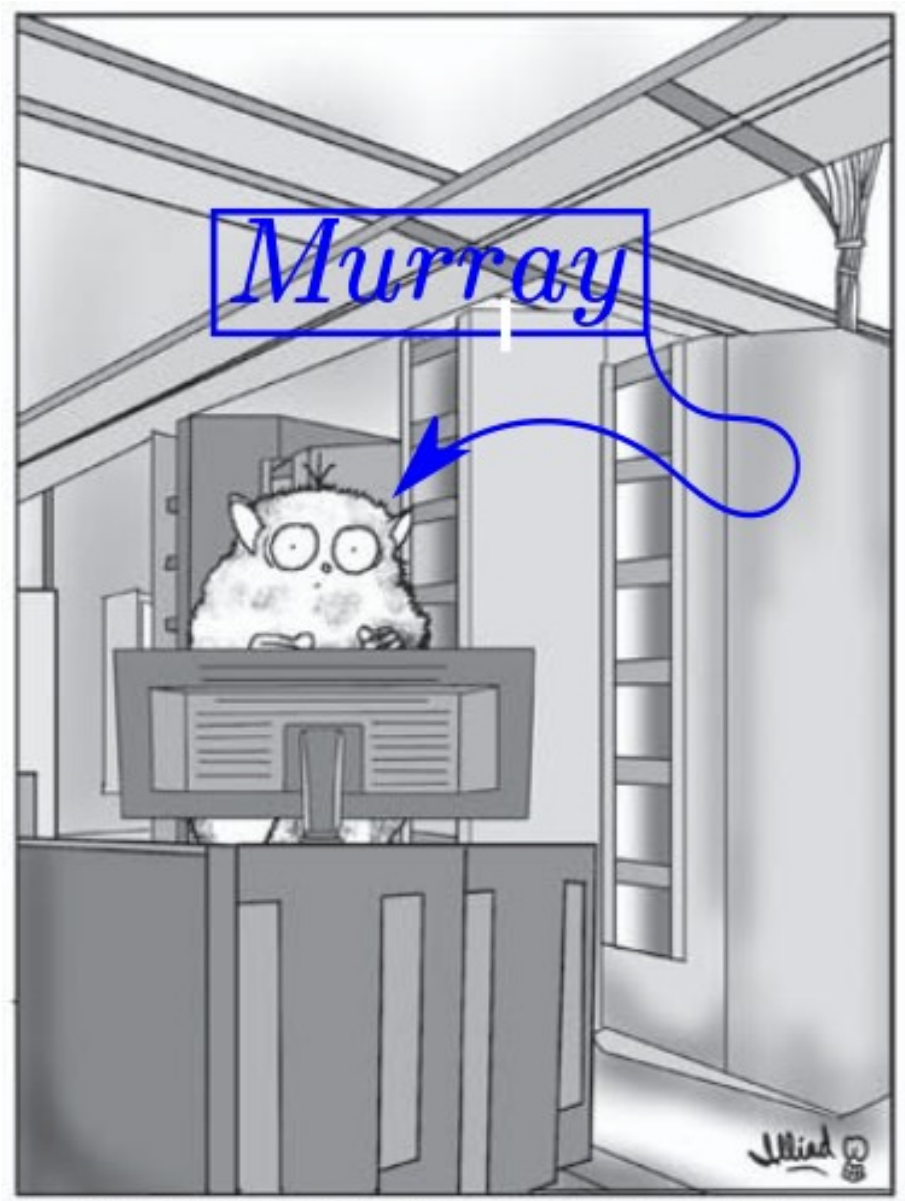
rm elimina los archivos especificados (directorios), no es posible recuperarlos.

cat concatena el contenido de varios archivos y lo muestra por pantalla. Puede ser combinado con **>** redireccionando la salida a un nuevo archivo.

more muestra el contenido del *fichero destino*, posee una función búsqueda **/.**

less similar a more pero con características adicionales como desplazar hacia atrás.

NOTA: Siempre se debe contar con los permisos necesarios!


Murray
SUDDENLY. MURRAY LEARNED THAT PABLO WAS ONLY JOKING WHEN HE SAID THAT "rm -rf /" WAS THE UNIX COMMAND FOR A DIRECTORY LISTING.

10. Enlaces a ficheros (direc/simbol)

- Los enlaces directos (duros) e indirectos (suave o simbólico) de un archivo o directorio a otro pueden ser creados usando el comando **ln**.
- **$ ln *filename linkname***
  - crea otra entrada de *filename*, digamos *linkname* (enlace duro). Ambas entradas son idénticas (ambos tienen ahora un número de enlace de 2).
  - Si alguno se modifica, el cambio se refleja en el otro archivo.
- **$ln –s *filename linkname***
  - crea un acceso directo llamado *filename* (*linkname* es un enlace blando). El acceso directo aparece como una entrada con un tipo especial (‘l’ ele).
- Se puede crear un enlace simbólico a un archivo que no existe, pero no un enlace duro.
- Se pueden crear enlaces simbólicos a través de diferentes dispositivos de disco físico o particiones, pero los enlaces duros están restringidos a la misma partición de disco.
- UNIX no suele permitir que los enlaces duros apuntan a directorios.

11. Múltiples nombres de archivo

- Múltiples nombres de archivo pueden ser especificados usando caracteres especiales de **búsqueda de patrones**. Las reglas son:
  - '**?**' coincide con cualquier carácter único en esa posición del nombre de archivo.
  - '*' Coincide con cero o más caracteres del nombre de archivo.
  - Caracteres encerrados entre corchetes “**[]**” coincidirá con cualquier nombre de archivo que tiene uno de esos caracteres en esa posición.
  - Una lista de cadenas separadas por comas y encerrada entre llaves “**{ }**” se expandirá como un producto cartesiano entre caracteres circundantes.

11. Múltiples nombres de archivo

- Por ejemplo:
  - **???** A todos los ficheros de tres caracteres en el directorio actual
  - **?ell?** nombres de archivo con cinco caracteres con “ell” en el medio
  - **el*** cualquier nombre de archivo que comienzan con “el”
  - **[m-z]*[a-l]** cualquier nombre de archivo que comience con una letra desde la “m” a la “z” y termine de la “a” a la “l”
  - **{/usr,} {/bin,/lib}/archivo** se expande a:
    /usr/bin/archivo, /usr/lib/archivo, /bin/archivo y /lib/archivo

- Tenga en cuenta que el shell de UNIX realiza estas expansiones (incluyendo cualquier coincidencia de nombre de archivo) de los argumentos de un comando *antes* de ejecutar el comando.

12. Comillas y caracteres especiales

- Como hemos visto ciertos caracteres especiales (por ejemplo, '*', '-', '{', etc.) son interpretados de una manera especial por el shell.
- Para pasar argumentos que utilizan estos caracteres a los comandos de forma directa, tenemos que utilizar *comillas*.
- Hay tres niveles que se pueden utilizar:
  - Trate de insertar un (\) frente al carácter especial.
  - Use comillas dobles (") alrededor de argumentos para prevenir la mayoría de las expansiones.
  - Use comillas simples simples (') alrededor de argumentos para prevenir todas las expansiones.
- Hay un cuarto tipo de *comillas* en UNIX. Las invertidas simples (`), se utilizan para pasar la salida de algún comando como un argumento de entrada a otro ➔.

12. Comillas y caracteres especiales

- Por ejemplo:

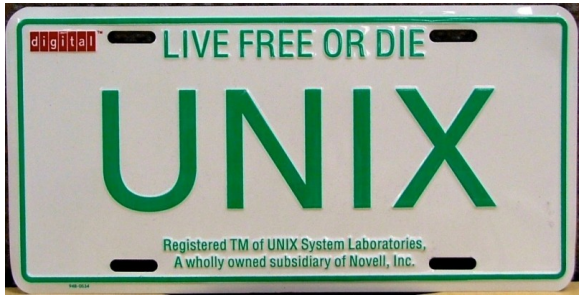
  $ **hostname**
  pepe

  $ **echo** esta máquina se llama `hostname`
  esta máquina se llama pepe

https://introunix.github.io/

- **Lecturas para curiosos (wiki++)**
- CAPÍTULO IV – UN SISTEMA DEL QUE DERIVARLOS A TODOS
  https://www.ionlitio.com/hackers-capitulo-iv/

- **CAPÍTULO V – UN PINGÜINO LLAMADO TUX**
  https://www.ionlitio.com/hackers-capitulo-v/



- **The Art of Unix Programming**
  http://www.faqs.org/docs/artu/index.html

  All the philosophy really boils down to one iron law, the hallowed 'KISS principle' of master engineers everywhere: