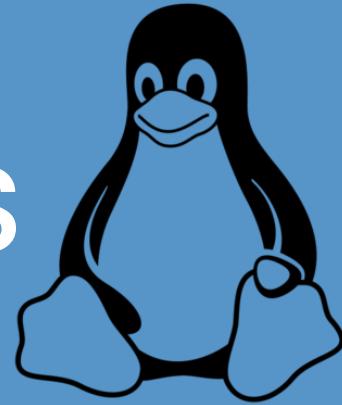


Sistemas Operativos

Módulo 2 - Clase Teórica



fundación
</argentic>

El Sistema Operativo



“Un sistema operativo (SO o, frecuentemente, OS —del inglés operating system—) es el software principal o conjunto de programas de un sistema informático que gestiona los recursos de hardware y provee servicios a los programas de aplicación de software, ejecutándose en modo privilegiado respecto de los restantes.”

Tanenbaum, A. (1992) Modern Operating Systems

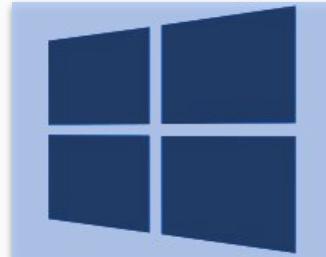
- 〔 Es el primer programa en ejecutarse
- 〔 Se encarga principalmente de gestionar los recursos de Hardware:
 - 〔 Procesador: gestiona el uso de CPU de los Procesos
 - 〔 Memoria: administra el espacio en memoria para alojar Procesos
 - 〔 Periféricos:
 - # Discos
 - # Tarjetas de Red
 - # Monitor
 - # Impresoras
 - # Mouse / Teclado

Principales funciones:

- **Gestionar la CPU:** Se encarga de administrar la CPU que va a estar repartida entre todos los procesos que se estén ejecutando.
- **Gestionar la RAM:** Para asignar el espacio de memoria a cada aplicación y a cada usuario, en caso de ser necesario. Cuando esta memoria se hace insuficiente, se crea una memoria virtual, de mayor capacidad, pero como está en el almacenamiento secundario (disco duro), es más lenta.
- **Gestionar el I/O:** El sistema operativo crea un control unificado de los programas a los dispositivos, a través de drivers.
- **Gestionar los procesos:** Se encarga de que las aplicaciones se ejecuten sin ningún problema, asignándoles los recursos que sean necesarios para que estas funcionen. Si una de ellas no responde, se procede a matar el proceso.



Ejemplos de Sistemas Operativos “conocidos”:

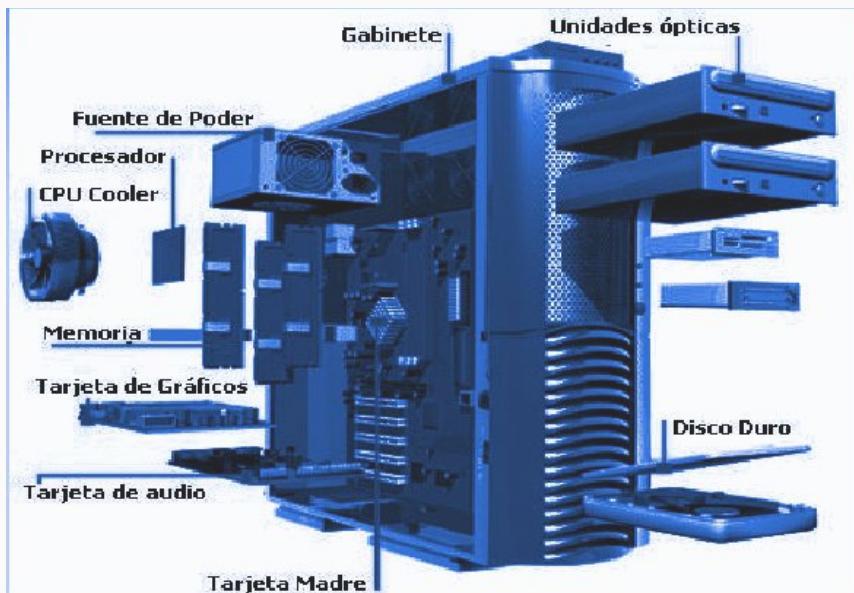


En el curso nos enfocaremos en el **Sistemas Operativo Linux**

Como todo programa “reside” en disco pero “vive” en la memoria principal y precisa CPU para “vivir”.

Reside = Está instalado

Vive = Se ejecuta

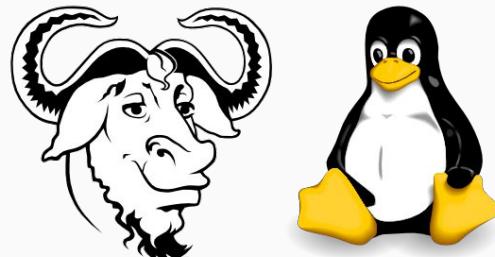


GNU/Linux

© 2007 - Copyleft 2007
Software Foundation Inc. <<http://www.gnu.org>>
Everyone is permitted to copy and
distribute verbatim copies of this
document, but changing it is not
allowed.
Assembly. The GNU General Public License
is a free, copyleft license for software
and other kinds of works. The licenses
for most software and other practical
works are designed to take away your
freedom to share and change the works.
By contrast, the GNU General Public
License is intended to guarantee your
freedom to share and change all versions
of a program, so that it remains
free software for all its users. The
Free Software Foundation, use the GNU
General Public License for most of our
software; it applies also to any other
works in this way by its authors.
You can't limit to your programs, too.
Because of free software, we are
able to have freedom, not price. Our
General Public Licenses are designed to
make sure that you have the freedom to
distribute copies of free software (and
charge for them if you wish), that you
receive source code or can get it if you
want it, that you can change the
software in new pieces of it to new free
programs, and that you know how you can do
these things. To protect your rights, we
need to prevent others from denying you
these rights or asking you to surrender
the rights. Therefore, you have certain
responsibilities if you distribute
the software. If you modify
the responsibility to respect the
rights of others. For example, if you
distribute copies of such a program
without gratis or for a fee, you must
give on to the recipients the same
rights that you received. You must
make sure that they, too, receive or can
get the source code. And you must show
them these terms so they know their
rights.

```
    notice("  Linux\n"
  setup_arch(  command_line\n"
  m_init_cpus(  init_main\n"
  setup_command(  command_line\n"
  setup_nr_cpu_ids()\n"
  boot_cpu_state_init()\n"
  mp_prepare_boot_cpu()\n"
  build_all_sonobooks(NULL, NULL)\n"
  page_alloc_init()\n"
  pr_notice("Kernel command line\n"
  parse_early_params()\n"
  after_dashes = parse_args("booting\n"
    static_command_line,\n"
    start_kernel = static\n"
    -1, NULL, &unknown_bootarg)\n"
  if (!IS_ENABLED(CONFIG_EARLY_DEBUG))\n"
    parse_args("Setting init args\n"
      NULL, set_init_arg)\n"
  jump_label_init()\n"
  setup_log_buf(0)\n"
  pidhuan_init()\n"
  vfat_caches_init_early()\n"
  __init_main_executable()\n"
  setup_init()\n"
  m_init()\n"
  sched_init()\n"
  preempt_disable()\n"
  if (WARN(!irq_desc))\n"
    "Interrupts were disabled\n"
    local_irq_disable()\n"
    idr_init_cache()\n"
    trace_init()\n"
    early_irq_init()\n"
    cpu_init_node()\n"
    timers_init()\n"
    timekeeping_init()\n"
    sched_clock_postinit()\n"
    printk(KERN_INFO "%s: perf counter\n"
    profile_init() call function\n"
    #if !IRQ_DISABLED\n"
    "Interrupts were disabled\n"
    early_boot_irqs_disabled = false\n"
    knew_cache_sonobooks()\n"
    kernel_init_freeable()\n"
    sync_synchronise_full()\n"
    free_initmem()\n"
    mark_as_stale(SYSTEM_BLOCK)
```

GNU/Linux, es el término empleado para referirse a la combinación del sistema operativo GNU, desarrollado por la FSF, y el núcleo(kernel) Linux, desarrollado por Linus Torvalds y la Linux Foundation. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera bajo los términos de la GPL (Licencia Pública General de GNU) y otra serie de licencias libres.



GNU/Linux

El proyecto GNU, que se inició en 1983 por Richard Matthew Stallman, tiene como objetivo el desarrollo de un sistema operativo completo compuesto enteramente de software libre.

En 1991 Linus Torvalds empezó a trabajar en un reemplazo no comercial para MINIX¹² que más adelante acabaría siendo Linux.



Cuando Torvalds presentó la primera versión de Linux en 1992, el proyecto GNU ya había producido varias de las herramientas fundamentales para el manejo del sistema operativo, incluyendo un intérprete de comandos, una biblioteca C y un compilador, pero como el proyecto contaba con una infraestructura para crear su propio núcleo (o kernel), el llamado Hurd, y este aún no era lo suficientemente maduro para usarse, se optó por utilizar Linux para poder continuar desarrollando el proyecto GNU, siguiendo la tradicional filosofía de cooperación entre desarrolladores.



¿QUÉ ES LINUX?



Es un **kernel** (núcleo de sistema operativo) y un conjunto de **sistemas operativos** de código abierto.



ORIGEN

Es el resultado de la fusión de dos proyectos inspirados en **Unix**, el primer sistema operativo portable.

GNU

RICHARD STALLMAN



Quería desarrollar un **sistema operativo libre**, pero le faltaba el **kernel**.



Linux es el SO más usado en supercomputadoras, Cloud Computing y Machine Learning. Empresas como **Microsoft**, **Facebook**, **Google** o **Adobe** son miembros de **The Linux Foundation**.



DISTRIBUCIONES LINUX MÁS USADAS

EMPRESAS



redhat

Red Hat Enterprise Linux (RHEL) es la **distro empresarial más importante** (no es gratis).



CentOS

Community ENTerprise Operating System. Es un **fork gratuito de Red Hat**.



Suse ofrece **soluciones de servidores**, desktop y sistemas embebidos para empresas.



USUARIO FINAL



KALI

Debian es una de las distros más completas, antiguas y estables de Linux



Basada en Debian, trae herramientas para **auditoría y seguridad informática**.



ubuntu

Basada en Debian, es la distro más famosa de Linux.



fedora

Distro para **usuario final basada en Red Hat**, pero desarrollada por la comunidad.

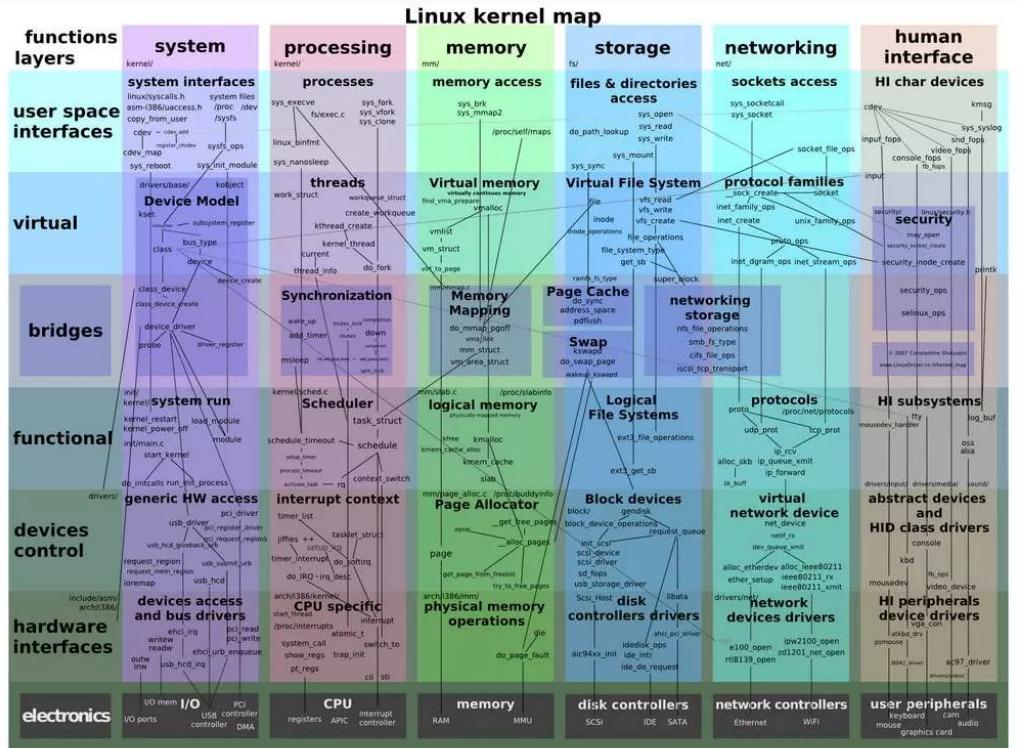


Linux Mint, basada en Ubuntu, intenta dar una experiencia cercana a Windows.

El Sistema Operativo Linux

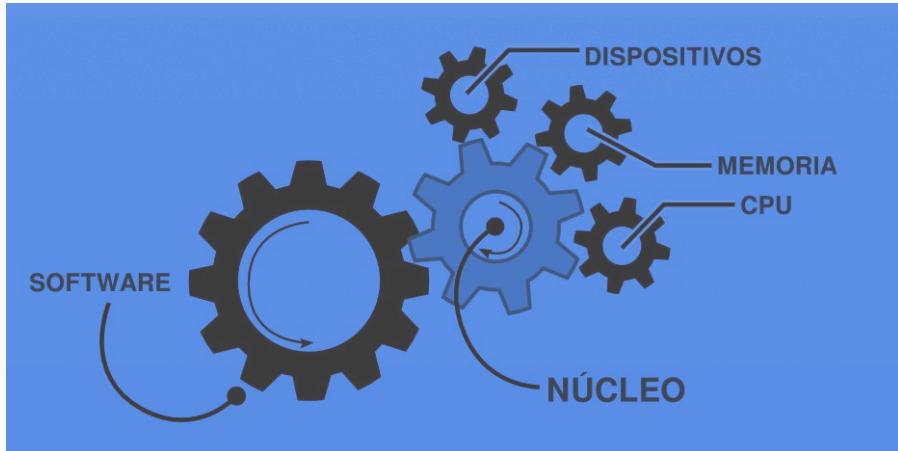
Los tres componentes fundamentales el Sistema Operativo son:

- [El Kernel (Núcleo)]
- [El Sistema de Archivos]
- [La Shell]



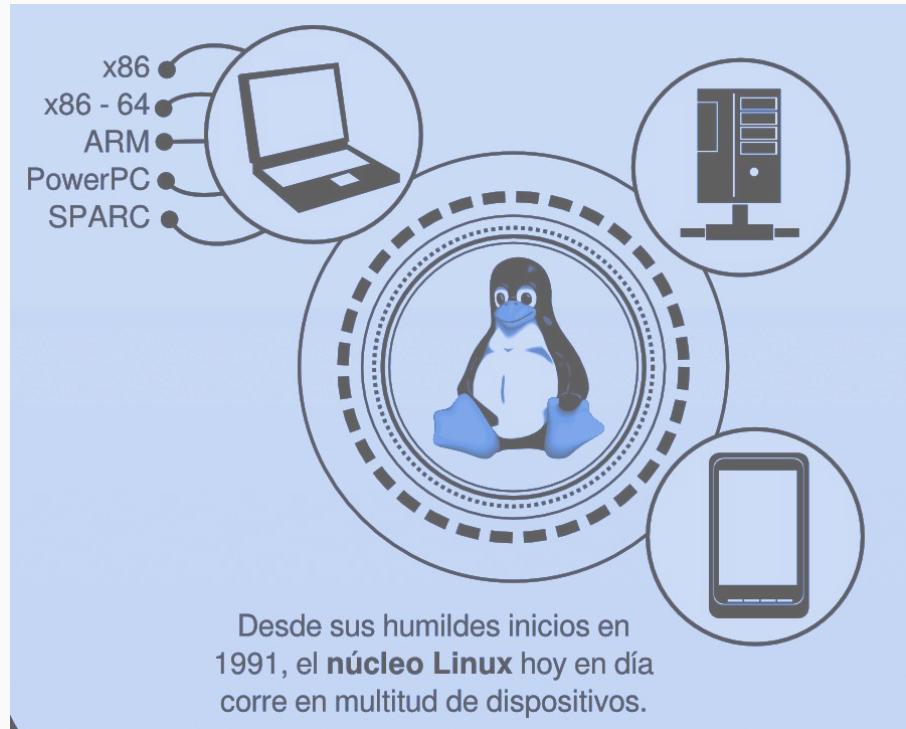
El Núcleo de un Sistema

Operativo es el componente que interactúa con el Hardware y administra la comunicación entre el Software y éste.



El Núcleo Linux soporta múltiples arquitecturas:

- [x86]
- [x86-64]
- [ARM]
- [PowerPC]
- [SPARC]



El Núcleo Linux está escrito mayormente en el lenguaje de programación C en conjunto con la colección de compiladores GNU GCC.



Las Versiones del Núcleo se componen de 3 partes:

- [Versión del Núcleo]
- [Revisión Mayor]
- [Revisión Menor]

```
[root@localhost ~]# hostnamectl
  Static hostname: localhost.localdomain
    Icon name: computer-vm
      Chassis: vm
    Machine ID: ac48eeceeb74c049ea6ecdd846ce3d5
        Boot ID: 02f2b3fb9fd140d3a8a984a12938f6e4
  Virtualization: kvm
Operating System: CentOS Linux 7 (Core)
      CPE OS Name: cpe:/o:centos:centos:7
          Kernel: Linux 3.10.0-957.21.2.el7.x86_64
    Architecture: x86-64
```

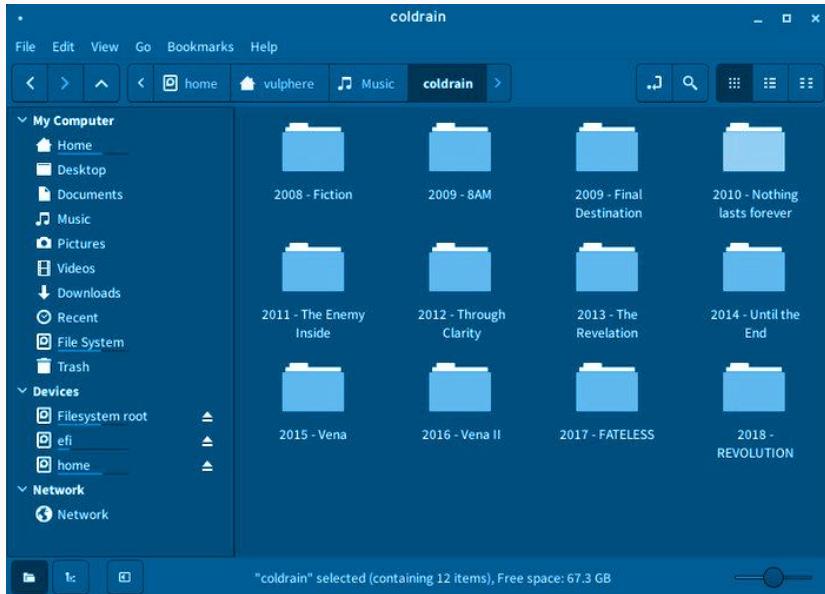
El Sistema de Archivos o sistema de ficheros es el componente del sistema operativo encargado de administrar y facilitar el uso de las memorias periféricas, ya sean secundarias o terciarias.

La mayoría de los sistemas operativos manejan su propio sistema de archivos (ext4, NTFS, ZFS)

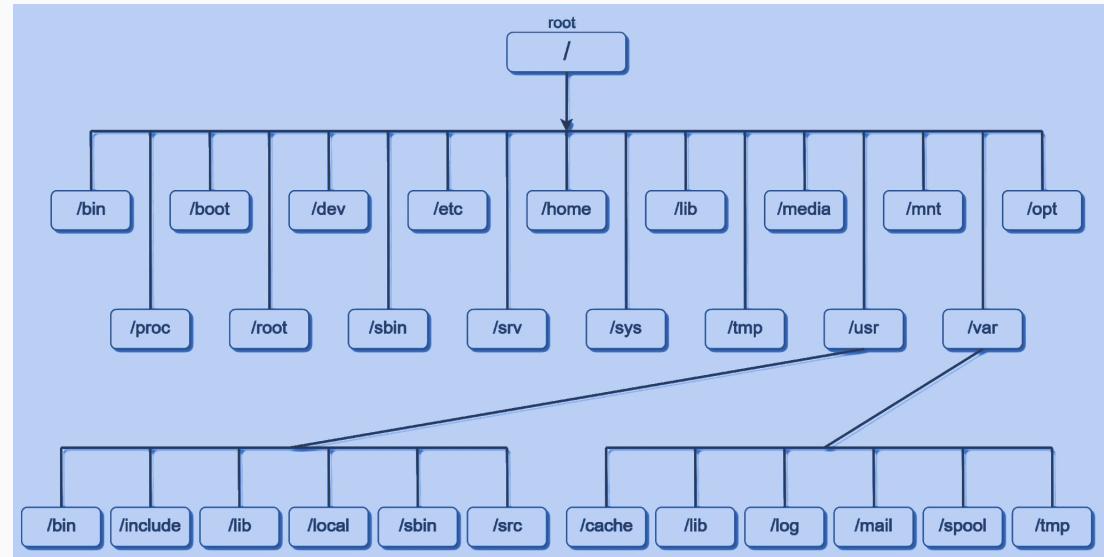


El Sistema de Archivos ~

Sus principales funciones son la asignación de espacio a los archivos, la administración del espacio libre y del acceso a los datos resguardados. Estructuran la información guardada en un dispositivo de almacenamiento de datos (normalmente un disco duro de una computadora), que luego será representada ya sea textual o gráficamente utilizando un gestor de archivos.



En Linux es muy importante entender cómo se estructura y ordena la jerarquía del Sistema de Archivos.

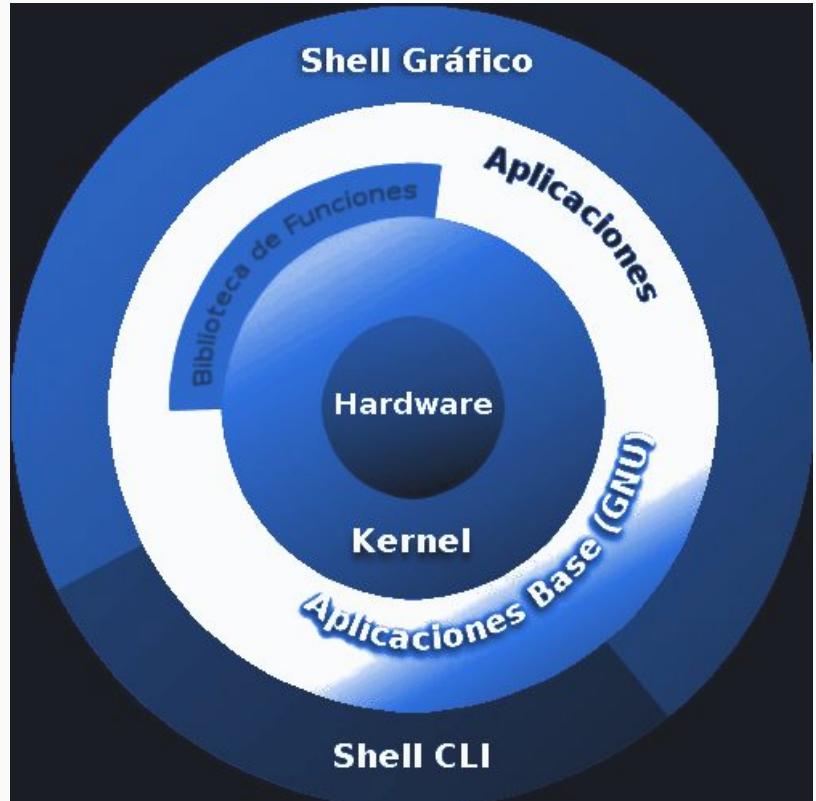


- # Seguridad o permisos
 - @ Listas de control de acceso (ACL)
 - @ UGO ("Usuario, Grupo, Otros", o por sus siglas en inglés: "User, Group, Others")
 - @ Capacidades granuladas
 - @ Atributos extendidos (ej.: sólo añadir al archivo pero no modificar, no modificar nunca, etcétera)
- # Mecanismo para evitar la fragmentación
- # Capacidad de enlaces simbólicos o duros
- # Integridad del sistema de archivos (Journaling)
- # Soporte para archivos dispersos
- # Soporte para cuotas de discos
- # Soporte de crecimiento del sistema de archivos nativo

La Shell o intérprete de comandos es un programa que provee una interfaz de usuario para acceder a los servicios del Sistema Operativo.

Dependiendo del tipo de interfaz que empleen, los shells pueden ser:

- # De líneas texto (CLI, Command-Line Interface, interfaz de línea de comandos),
- # Gráficos (GUI, Graphical User Interface, interfaz gráfica de usuario),
- # De lenguaje natural (NUI, Natural User Interface, interfaz natural de usuario).



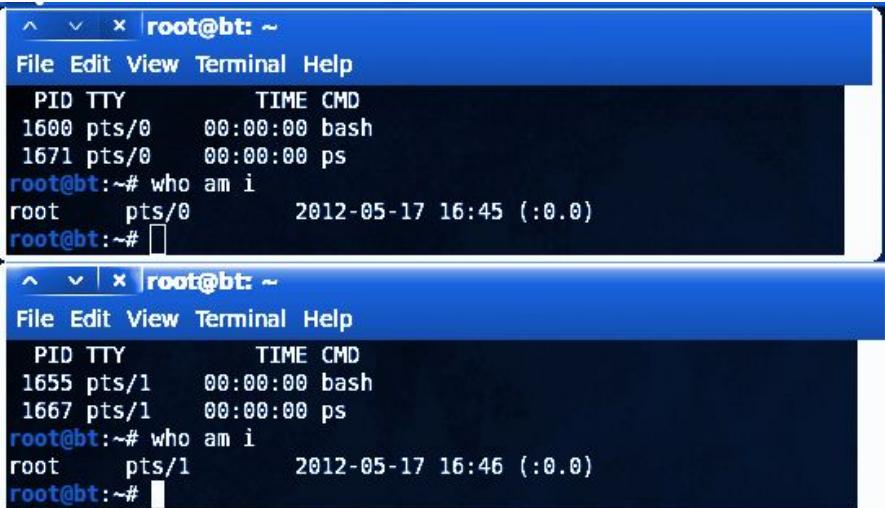
Las Shell son necesarios para invocar o ejecutar los distintos programas o funciones del Sistema Operativo.

Ejemplos de Shells (CLI):

```
# Símbolo del sistema (Windows)
# Almquist Shell (Ash)
# Bourne Shell
# Bash (Bourne Again Shell)
# Fish
# Korn Shell
# Zsh
```

La Terminal no es lo mismo que la Shell. Es el programa a través del cual accedemos al Shell para poder ejecutar comandos.

Existen distintos tipos de terminales, se pueden ejecutar varias terminales al mismo tiempo por usuario y se pueden configurar para la comunidad de cada uno.



The image displays two separate terminal windows side-by-side. Both windows have a blue header bar with the text "root@bt: ~" and a menu bar below it containing "File", "Edit", "View", "Terminal", and "Help".

Top Terminal Window:

```
^ v x | root@bt: ~
File Edit View Terminal Help
PID TTY      TIME CMD
1600 pts/0    00:00:00 bash
1671 pts/0    00:00:00 ps
root@bt:~# who am i
root    pts/0    2012-05-17 16:45 (:0.0)
root@bt:~# [ ]
```

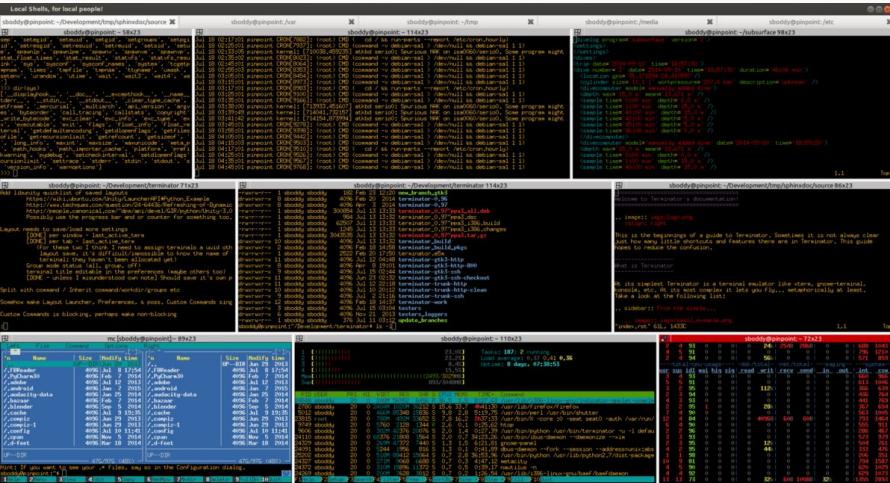
Bottom Terminal Window:

```
^ v x | root@bt: ~
File Edit View Terminal Help
PID TTY      TIME CMD
1655 pts/1    00:00:00 bash
1667 pts/1    00:00:00 ps
root@bt:~# who am i
root    pts/1    2012-05-17 16:46 (:0.0)
root@bt:~# [ ]
```

Existen distintos tipos de terminales, se pueden ejecutar varias terminales al mismo tiempo por usuario y se pueden configurar para la comunidad de cada uno.

Algunos ejemplos de terminales de Linux:

- # GNOME Terminal
- # Konsole
- # Terminator
- # Lxterminal



Virtualización



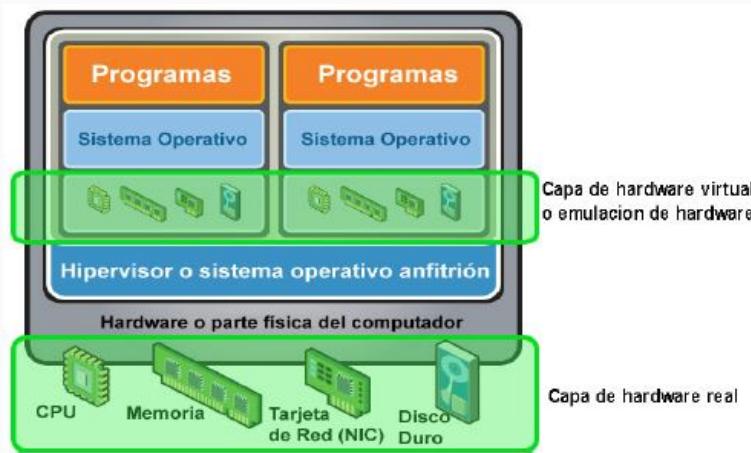
La Virtualización es una capa abstracta que desacopla el hardware físico del sistema operativo para brindar una mayor flexibilidad y utilización de los recursos de TI. El computador físico comparte recursos para crear las máquinas virtuales (VM).

La virtualización mejora los recursos de hardware que se utilizan en el centro de datos. Por ejemplo, en lugar de ejecutar un servidor en un sistema informático, se puede crear un grupo de servidores virtuales en el mismo sistema informático, al utilizar y devolver servidores al grupo según sea necesario.



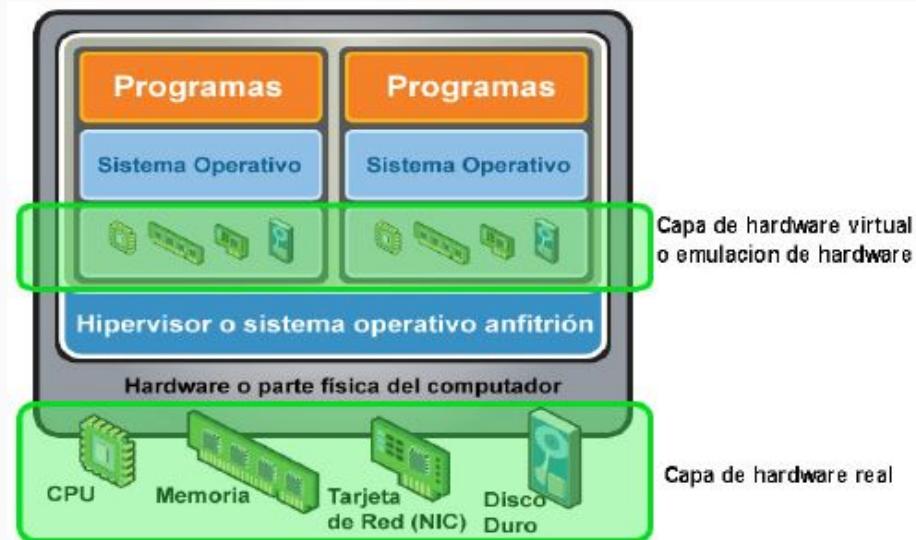
Virtualización del Sistema Operativo (Máquinas Virtuales)

- Separación entre SO y el hardware real.
- SO Anfitrión (Host) SO Alojado (Guest)
- Permite instalar y usar distintos sistemas operativos instalados en los “hardwares” Alojados con el hardware del SO Anfitrión.



El programa/software Hipervisor permite “emular” un hardware virtual para destinar los recursos necesarios (CPU, Memoria RAM, Disco y Periféricos) a la cantidad de Máquinas Virtuales que nos permita crear.

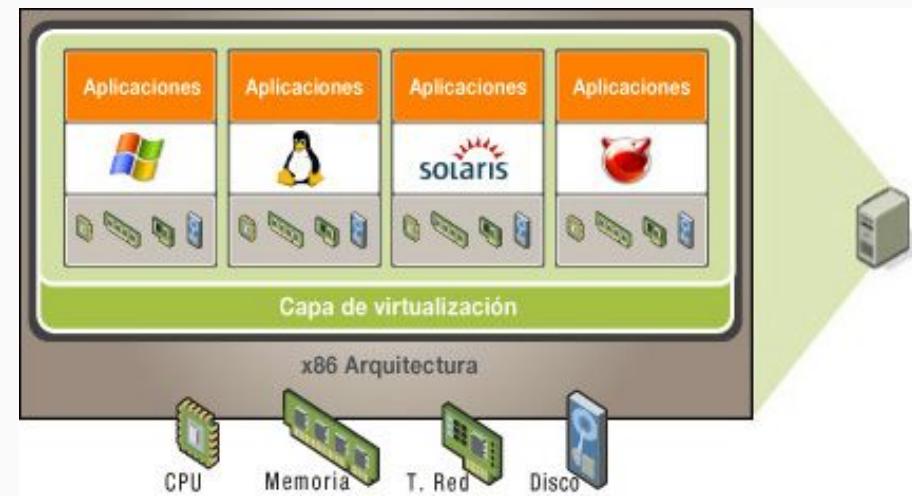
A mejor Hardware Físico
Mayor es la posibilidad de ofrecer
hardware emulado a las VMs
(Virtual Machines).



En cada VM creada se podrá instalar el Sistema Operativo deseado para realizar pruebas, instalar software, librerías o servicios específicos que solo corren en ese tipo de Sistema Operativo.

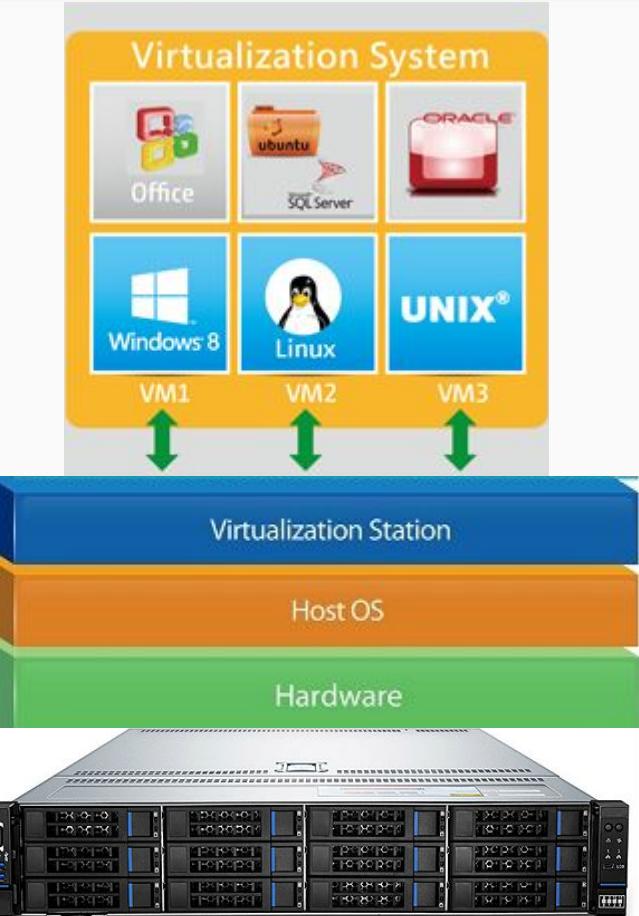
Ejemplo:

Puedo tener Windows 10 como mi SO Guest instalado “nativo” en mi PC, contar con un Software Hipervisor que me permita destinar 4 GB de Memoria RAM, 2 núcleos de mi CPU y 100 GB de almacenamiento en disco a un Sistema Operativo Linux instalado en una VM.

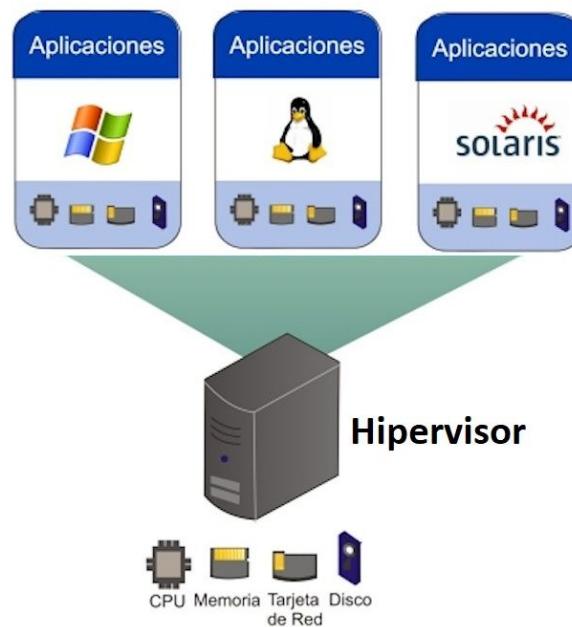


De ésta forma es posible que convivan distintos servicios o programas de distintos Sistemas Operativos en un mismo hardware lo que brinda:

- Optimización de Recursos.
- Eficiencia Energética.
- Facilidad de Administración.
- Aislamiento y Seguridad.
- Recuperación y Respaldo.
- Escalabilidad.
- Implementación Rápida de Servidores.
- Compatibilidad y Flexibilidad.
- Ahorro de Costos.
- Desarrollo Ágil y DevOps.



Un hipervisor es un software que permite la virtualización de recursos de computación, como la CPU, la memoria, el almacenamiento y los dispositivos de entrada/salida. También se conoce como monitor de máquina virtual (VMM, por sus siglas en inglés).

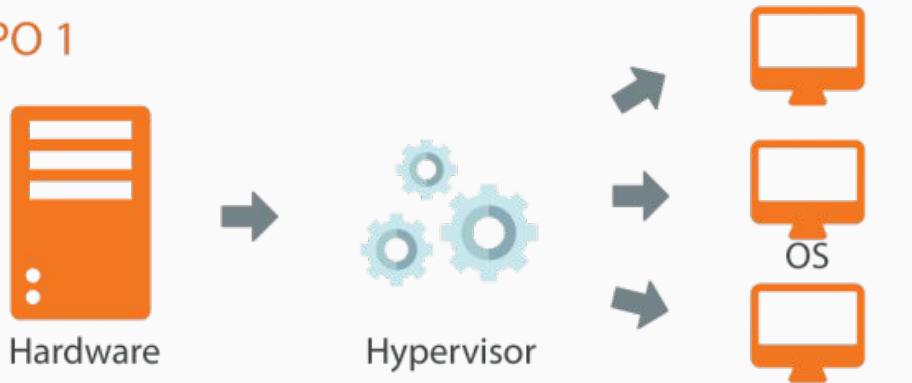


El propósito principal de un hipervisor es crear y ejecutar múltiples máquinas virtuales (VM) en un único hardware físico. Esto permite que varios sistemas operativos y sus aplicaciones se ejecuten de forma independiente, como si estuvieran funcionando en hardware físico separado, aunque en realidad comparten los recursos de una máquina física subyacente.



Hipervisores de tipo 1 (nativo o bare-metal): Estos hipervisores se ejecutan directamente sobre el hardware físico, sin necesidad de un sistema operativo anfitrión adicional. Típicamente, se encuentran en servidores de virtualización y proporcionan un alto rendimiento y escalabilidad. Ejemplos de hipervisores de tipo 1 incluyen Proxmox VE, Xen, Kernel-based Virtual Machine (KVM), Microsoft Hyper-V, VMware ESXi, Oracle VM Server.

TIPO 1



Hipervisores de tipo 2 (hosted): Estos hipervisores se ejecutan sobre un sistema operativo anfitrión tradicional. Requieren un sistema operativo anfitrión como Windows, macOS o Linux para funcionar. Son comúnmente utilizados en entornos de desarrollo y pruebas, así como en sistemas de escritorio para virtualización de aplicaciones. Ejemplos de hipervisores de tipo 2 incluyen VMware Workstation, Parallels Desktop, VirtualBox, VMware Player, QEMU y Bhyve..

TIPO 2



OS



Hypervisor



|

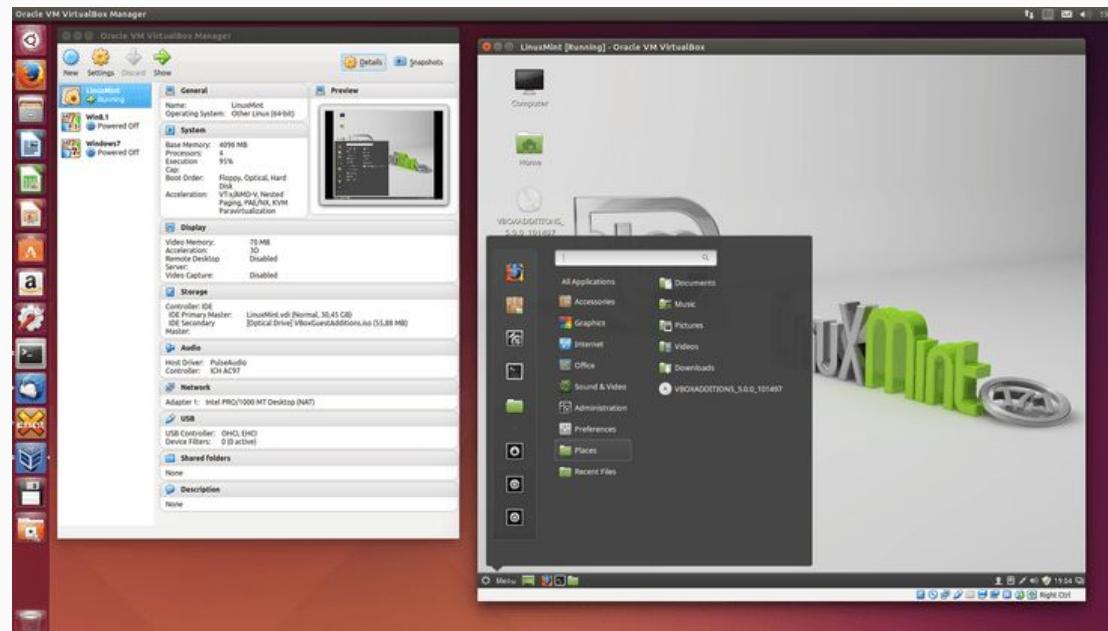


VirtualBox



Oracle VM VirtualBox (conocido generalmente como VirtualBox) es un software de virtualización para arquitecturas x86/amd64. Actualmente es desarrollado por Oracle Corporation como parte de su familia de productos de virtualización..

Por medio de esta aplicación es posible instalar sistemas operativos adicionales, conocidos como «sistemas invitados», dentro de otro sistema operativo «anfitrión», cada uno con su propio ambiente virtual



Las distribuciones de Linux, comúnmente conocidas como "distros", son variantes del sistema operativo Linux que incluyen un núcleo de Linux, software de sistema básico, herramientas de administración y paquetes de software específicos. Cada distribución de Linux está diseñada para satisfacer diferentes necesidades y preferencias de los usuarios.

Aunque todas las distribuciones de Linux comparten el mismo núcleo (el kernel de Linux), difieren en cuanto a la selección de software, la configuración predeterminada, la filosofía de diseño, la gestión de paquetes y el enfoque para satisfacer las necesidades de los usuarios.



Debian es una distribución de sistema operativo Linux de código abierto y gratuito, conocida por su estabilidad, seguridad y compromiso con el software libre. Es una de las distribuciones más antiguas y ampliamente utilizadas en el mundo del software de código abierto.

Algunas características importantes de Debian son:

Estabilidad: Debian se enfoca en ofrecer una plataforma estable y confiable para los usuarios. Sus lanzamientos se basan en pruebas exhaustivas y se centran en la estabilidad a largo plazo.

Software libre: Debian se adhiere estrictamente a los principios del software libre y promueve la filosofía del código abierto. Todos los paquetes de software incluidos en Debian deben cumplir con los criterios de la Licencia Pública General de GNU (GPL) o una licencia similar.

Gestión de paquetes: Debian utiliza el sistema de gestión de paquetes "APT" (Advanced Package Tool), que facilita la instalación, actualización y eliminación de software en el sistema. APT gestiona las dependencias de los paquetes de forma automática, lo que simplifica el mantenimiento del sistema.

Arquitecturas y portabilidad: Debian es compatible con múltiples arquitecturas de hardware, incluyendo x86, x86_64, ARM, MIPS, PowerPC, entre otras. Esto hace que Debian sea adecuado para una amplia variedad de dispositivos y entornos.

¿Preguntas?