

①

linux

Kernel : El controlador controla el hardware

GNU / Linux : Combinación de herramientas de software libre (GNU) + el Kernel

Unix - like : Linux no es unix (mucha registrada), pero se comporta y sigue sus estándares

Portabilidad : Escrito en C, lo que permite su ejecución en servidores smartphones y dispositivos IoT

② El ecosistema de distribuciones

- Componentes distro
- Kernel de Linux
- Herramientas GNU
- Gestor de paquetes
- Entorno de escritorio

③ open source y licenciamiento

- Código fuente : disponible para ver, modificar y redistribuir
- Colaboración Global : desarrollo acelerado por la Comunidad

- ventaja benefici : eliminación de logros negros, control total sobre los procesos del sistema

#### ④ Línea de comandos (CLI) vs GUI

- Profundidad en cuenca de datos
- Eficiencia : menor consumo de recursos
- Automatización : Creación de scripts para tratos repetitivos (ETLs, limpieza de logs)
- Consistencia : Comandos idénticos en casi cualquier distro

GUI : Util para visualizaciones, pero secundaria en administración de servidores de datos.

#### ⑤ Conceptos clave para reposo

- Repo : servidores donde se aloja el software oficial de la distro

- 6 estds de poquitos : herramientas para instalar / actualizar software
- Shell : interprete de comandos que actua como interfaz entre el usuario y el kernel

# El sistema operativo como Gestor

- Multitareas: Capazidad de compartir el procesador entre múltiples tareas simultáneamente
- Servicio estandar: provee interfaces para que los programas soliciten recursos ( impresión, red, etc )
- Tipos por rol
  - escritorio: optimizado para GUI y productividad del usuario
  - Servidor: optimizado por acceso remoto, CLI y eficiencia de recursos ( sin interfaz gráfica en memoria )

## ② Ciclo de vida y estabilidad

- Ciclo de lanzamiento: frecuencia con la que aparecen nuevas versiones

• Ciclo de mantenimiento : tiempo que el proveedor ofrece actualizaciones de seguridad

versiones

Beta : Versiones nuevas, no probadas  
riesgo de fallo (ideal para desarrollo)

Estable : probado en el campo ; recomendada para producción

LTS : Versiones con soporte garantizado  
el 5 año o más (Ubuntu LTS)

### ③ Distro de Linux

- Family red hat

• RHL : largo, soporte corporativo,  
ciclo largo

• Fedora : El lab de red hat , software  
muy reciente

• Cent OS / rocky linux : versiones  
equivalentes compatibles con RHEL

- Familia debian

- debian : enfocada en estabilidad y software libre pura

- ubuntu : la mas popular ; excelente entre facilidad y potencia

- linux mint : Basado en ubuntu, optimizado para escritorio

- Familia SUSE

- SLES : version empresarial

- openSUSE : Version comunista

## Caso especiales

- Android : Kernel linux + marco virtual Dalvik , No GNU/ linux ( incompatible con apps de escritorio )

- Raspbian : optimizado para raspberry pi ( Educacion / IoT )

- Linux from scratch (LFS)  
proyecto para construir un SO  
desde cero (aprendizaje puro)

## 6) La interfaz : CLI vs GUI

- Terminal : la operación que abre  
la ventana de texto
- Shell : el intérprete que traduce los  
comandos o instrucciones del  
SO
- MOTD : mensaje informativo que  
aparece al iniciar sesión en un  
servidor vía CLI



# Aplicaciones y procesos

- Kernel como arbítrio : Gestiona memoria CPU y disco
- Abstracción de procesos : una operación es un conjunto de procesos monitoreados por el Kernel
- Multitarea : el Kernel altera las tareas tan rápido que parecen simultáneas

## ② Ecosistemas de servidores

- Web servers
- Apache HTTPD : robusto, modular el endpoint clásico
- NGINX : enfocado en alto rendimiento y concurrencia
- Base de datos relacionales
  - MySQL / MariaDB
  - PostgreSQL : preferido en ciencia de datos por su soporte avanzado de tipos de datos y extensión geográfica

- Infraestructuras
  - DNS (Bind) : Trasduce nombres a IPs
  - DHCP : Asigna IPs dinámicas en la red local
  - LDAP : directorios de usuarios y permisos (Similares a Active Directory)

### (3) Herramientas de consola y editores

- Shell
  - Bash : Estándar de la industria
  - Zsh / Fish : alternativas modernas con autocompletado
- Editores de texto
  - Vim
  - Nano

## (4) Gestion de paquetes

Debian	.deb	dpkg	apt / apt-get
RHEL / centos	.rpm	rpm	dnf / yum

## (5) lenguajes de programación en ciencia

- Compilados : C / C++ : velocidad máxima  
el kernel y los librerías base  
de datos de python escritos  
en C
- interpretados (Python y R) :  
flexibilidad e ideales para scripts y  
automatización
- Java : Ejecuta sobre una máquina  
virtual de java crucial para  
Big Data

## ⑥ Seguridad y privacidad

- RDT : Supervisorio
  - SSH siempre desencriptado
- Corafuego - Upables : herramientas de bajo nivel
- UFW : interfaz simplificada de ubuntu
- Encriptaciones : HTTPS para datos en tránsito ; VPN para túneles seguros entre almacenes y Oficina Central

## ⑦ Virtualización y nube

- Hipervisores : Software que gestiona máquinas virtuales
- Módulos de Nube : AWS, google Cloud, Azure
- Proveedor : infraestructura propia con mayor control

- Hibridos : mezcla de ambos  
pose escalar pisos de demonio
- Contenedores (Docker) mas ligeros  
que los VMs . No emulan  
hardware , comparten el Kernel  
del hosts



# Ciclo del software

- Código fuente: instrucciones legibles por seres humanos
- Compilación: proceso de traducir código fuente a instrucciones de máquina
- Interpretación: el código se ejecuta línea a línea por un intérprete  
Python, Bash

## ② Organización Clove

- FSF: fundada por Richard Stallman.  
Define las "4 libertades" del software  
Creadores de la licencia GPL
- OSI: Enfoque más empresarial y menos político. Validan si una licencia es realmente open source
- Creative Commons: licencias para contenido no-software (textos, imágenes, etc.)

BY (Attribution): de los creadores

SA (Share Alike): Compartir igual

NC (Non-Commercial): no uso comercial

ND (No-Derivatives): no modificar

### 3) Estándares y Compatibilidad

Posix : Estándar que asegura que las aplicaciones sean compatibles entre diferentes sistemas Unix-like (Linux, mac OS, BSD)

### 4) Modelos de negocio open source

¿Cómo regresar con esto?

- Soporte y garantía : Venta de conocimiento técnico
- Hardware : venta de equipos que corren linux
- SaaS (Software as a service) : Cobrar por el alojamiento del software en la nube
- Dual licensing : una versión gratuita (GPL) y la versión comercial para empresas que no quieren compartir su código

### ⑤ Conceptos técnicos para data science

- Tivoizolion : provisión de unos software libre <sup>en</sup> hardware bloqueado de evitando la tesisiva
- Floss : Free/libre open source software
- Public domain : El autor renuncia a todos los derechos



# El prompt y la shell

- Shell : intérprete de comandos
- prompt : indicador visual de espero entrada. Escribiré punto.

Símbolo ~ : representa el directorio Home del user

## ② Estructura de comando

- Formato : comando [ options ] [ argumentos ]
- opciones : Modificaron el comportamiento (`ls -l`) para listas largas `ls -h`
- argumentos : El objetivo del comando
- Case-sensitivity : Linux distingue estrictamente entre mayúsculas y minúsculas

### ③ History y producutors

! ! : Ejecuta el ultimo comando

! M : Ejecuta el comando M de historial

! -M : Ejecuta el m -esimo comando desde el final

! comando : ejecuta la version mas reciente de dicho comando

### ④ Variables en Bash

- Locales : Solo viven en la session actual . Se definen : nombre = valor
- Globales (De entorno) : disponibles para sub - procesos . Se crean con export nombre = valor

Variabilizaciones : Usar echo \$ nombre variable

Variable \$PATH : lista de directorios (repositorios) por : ) por donde la shell busca comandos

## 5) Tipos de comandos

- internal (Built-in) parte del código de la shell (cd, type)
- External : programas independientes en disco (ls, cat)
- Alias : Apodos para comandos largos
- Funciones : Bloques de código reutilizables

nombre - función () {

Comando 1

Comando 2

}

## 6) Comillas y caracteres de escape

- Comillas dobles ("") : Evitar la interpretación de variables y comando
- Comillas simples ('') tratan todo como texto literal, no interpreta nada

- Backslash (\) : Escapa en solo caracteres (ej: \\$ para que no sea una variable)
- Backticks (`) / \$() : ejecutan comandos dentro de otro (Sustitucion de comando)
- Ejemplo : echo hoy es \$(date)

Operador	Nombre	Lógica de Ejecución
:	Semicolon	Ejecuta cmd1 , luego cmd2 (sin importar si cmd1 falló).
&&	Logical AND	Ejecuta cmd2 solo si cmd1 fue exitoso (exit code 0).
	Logical OR	Ejecuta cmd2 solo si cmd1 falló (exit code != 0).