Estructura de los sistemas operativos

César Pedraza

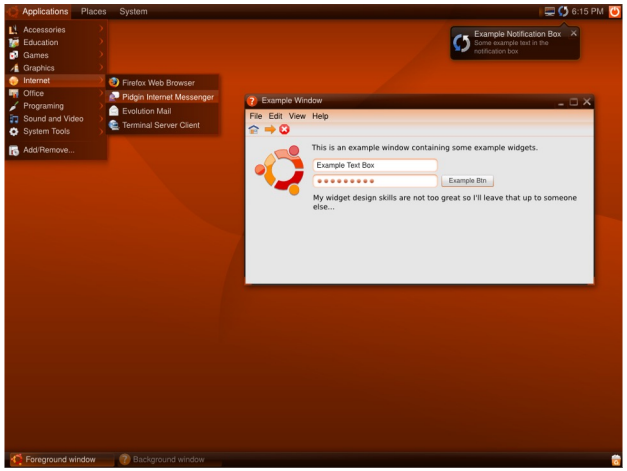
Introducción general

Un sistema operativo (SO) es el componente esencial que actúa como intermediario entre el usuario y el hardware. Su estructura define la forma en que se organizan sus componentes internos, cómo se relacionan entre sí y de qué manera ofrecen servicios. Una arquitectura bien diseñada permite mantener el sistema, facilitar su evolución y mejorar la seguridad y eficiencia.

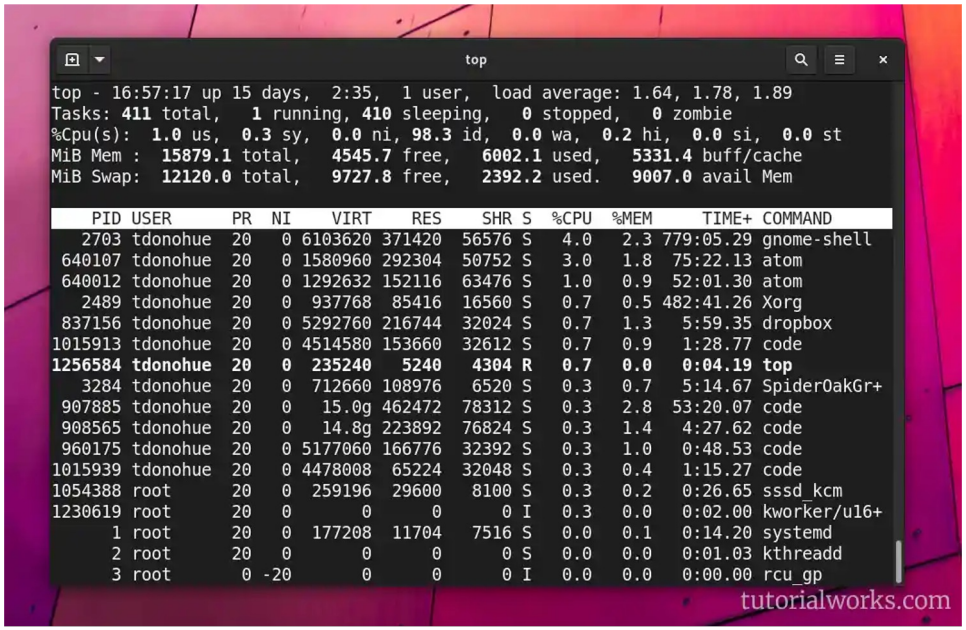
Servicios del SO (1) - Interfaz de usuario

La interfaz de usuario puede ser:

- Línea de comandos (CLI): interfaz textual donde el usuario ejecuta comandos.
- Interfaz gráfica (GUI): permite una interacción más visual y amigable mediante ventanas, menús y dispositivos apuntadores. El SO actúa como puente entre las solicitudes del usuario y los recursos del sistema.



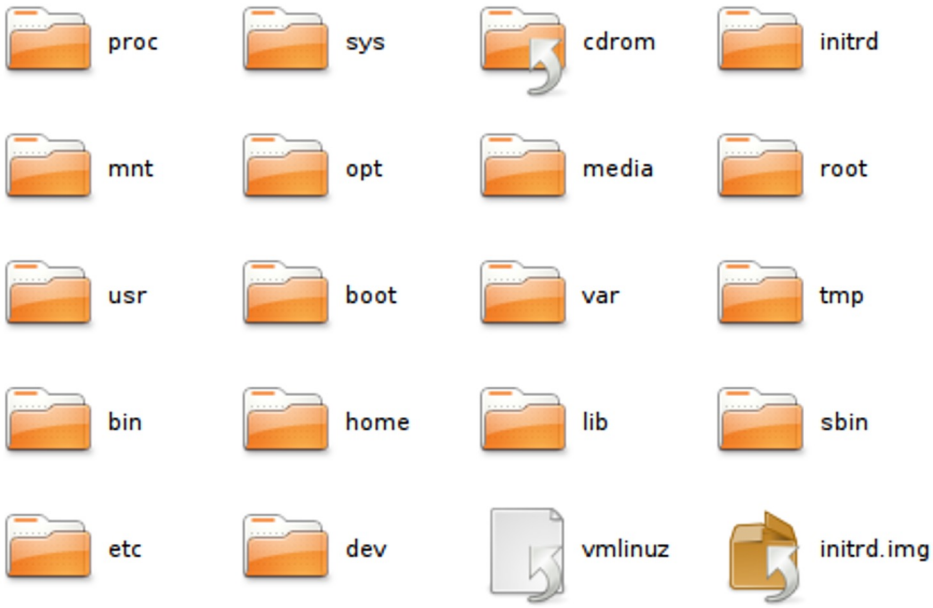
Servicios del SO (2) - Manejo de procesos



- Creación, eliminación y sincronización de procesos.
- Planificación del uso del CPU entre procesos activos.
- Control de estados del proceso (ejecución, espera, listo).
- Soporte para procesos concurrentes e hilos de ejecución.

Servicios del SO (3) - Manejo de archivos

- Abstracción para el acceso a datos.
- Operaciones: creación, apertura, lectura, escritura, cierre, eliminación.
- Estructuras jerárquicas de directorios.
- Control de acceso: permisos de lectura, escritura, ejecución.



Servicios del SO (4) - Entrada y salida

pxhere.com

- Administración de dispositivos heterogéneos.
- Manejo de controladores (drivers).
- Buffering, caching y spooling para mejorar el rendimiento.
- Proporciona abstracciones uniformes para dispositivos muy diversos.

Servicios del SO (5) - Comunicaciones

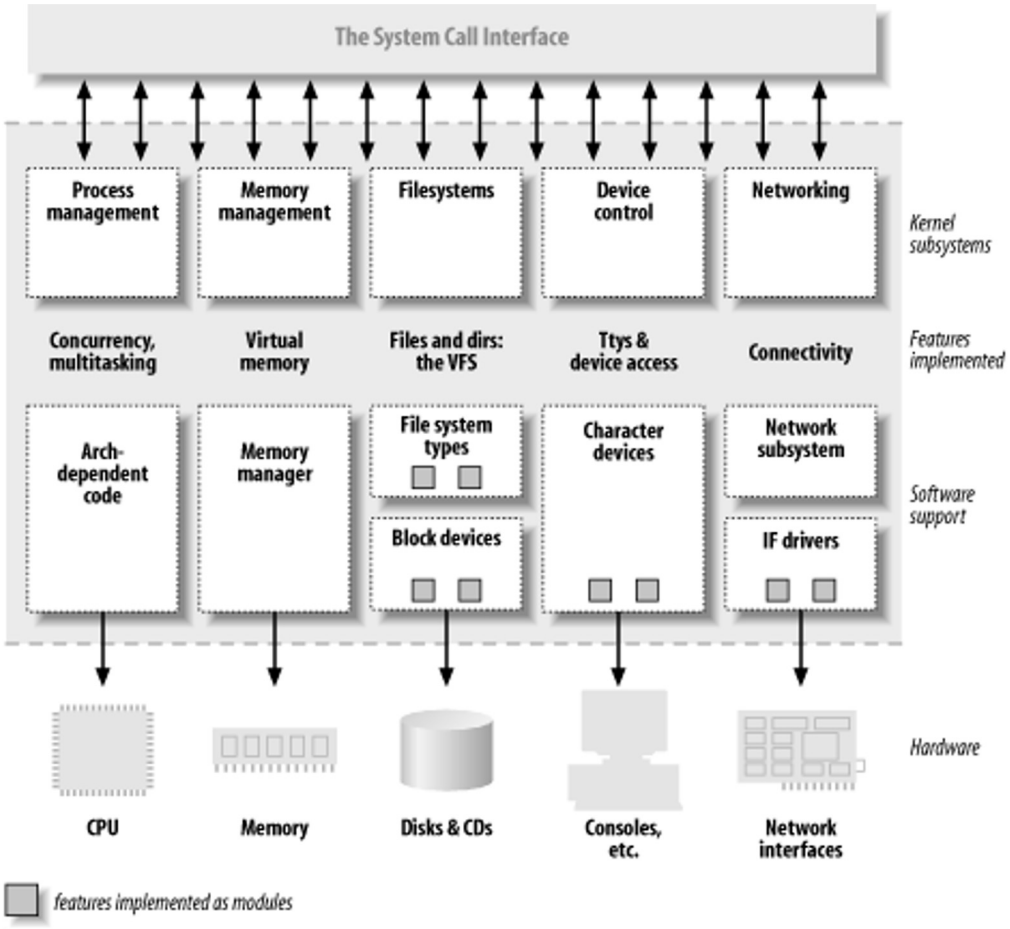
- Comunicación entre procesos (IPC), ya sea:
  - En memoria compartida
  - Por paso de mensajes
- Comunicación entre procesos en diferentes máquinas.
- Soporte a protocolos de red, RPC (Remote Procedure Call), y sockets.

Servicios del SO (6) - Seguridad

- Autenticación de usuarios.
- Control de acceso a archivos y dispositivos.
- Mecanismos de aislamiento entre procesos.
- Registro de auditorías y detección de actividades sospechosas.

Llamadas al sistema - Concepto

- Son el punto de entrada para que un programa de usuario acceda a los servicios del sistema operativo.
- Implican un cambio de modo: de modo usuario a modo núcleo (privilegiado).
- Se implementan vía interrupciones o instrucciones especiales.



Llamadas al sistema - Interfaz

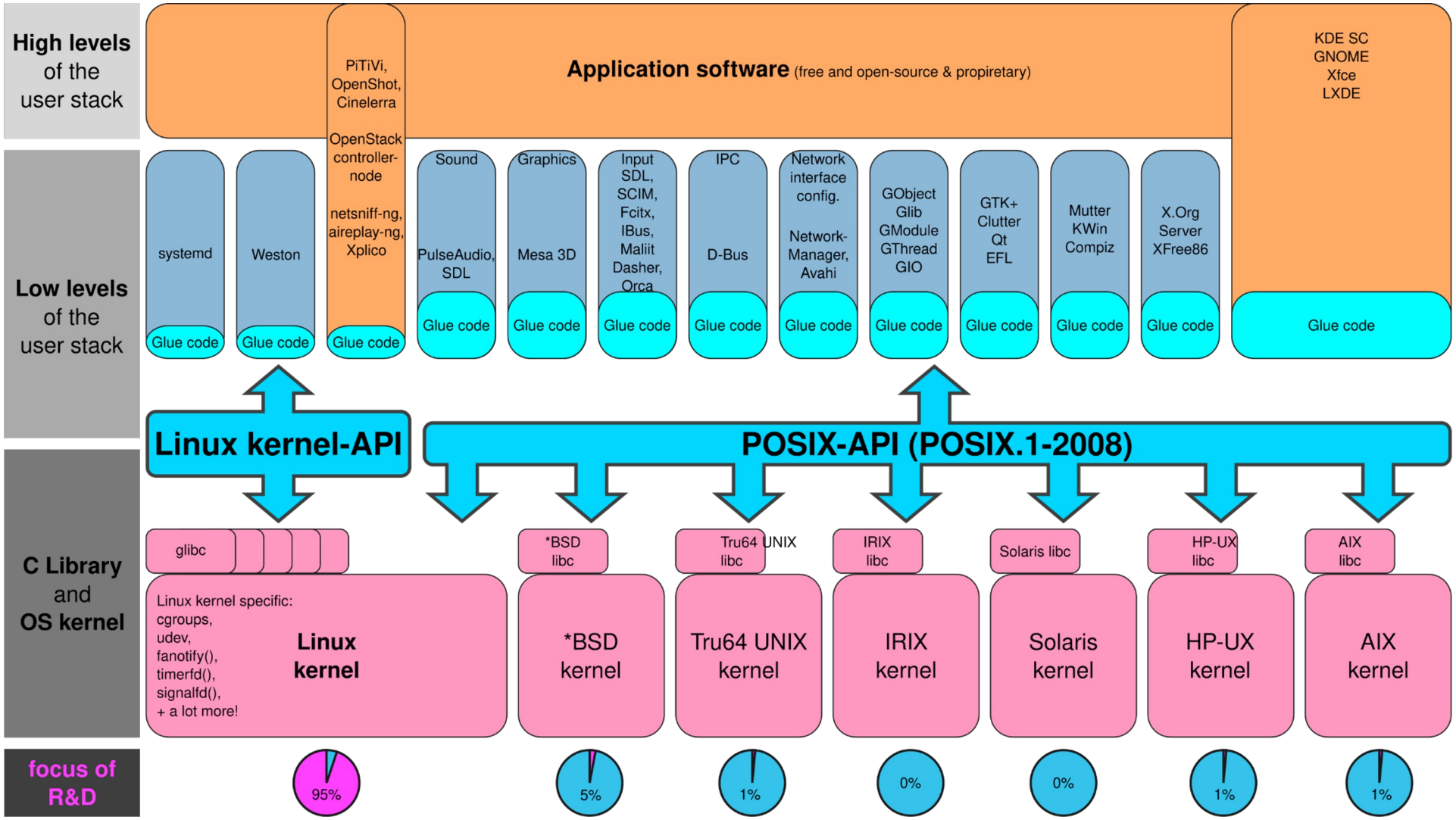
- Las aplicaciones invocan funciones definidas en una API (por ejemplo POSIX, WinAPI).
- Las API traducen estas funciones en llamadas al sistema reales.
- Ejemplo: `open("archivo.txt")` se convierte en una syscall de apertura de archivo.

Tipos de llamadas al sistema (1) - Procesos

- Creación y finalización de procesos (`fork`, `exit`).
- Carga y ejecución de programas (`exec`).
- Espera por eventos (`wait`).
- Manejo de prioridades y señales entre procesos.

Tipos de llamadas al sistema (2) - Archivos - Sistema

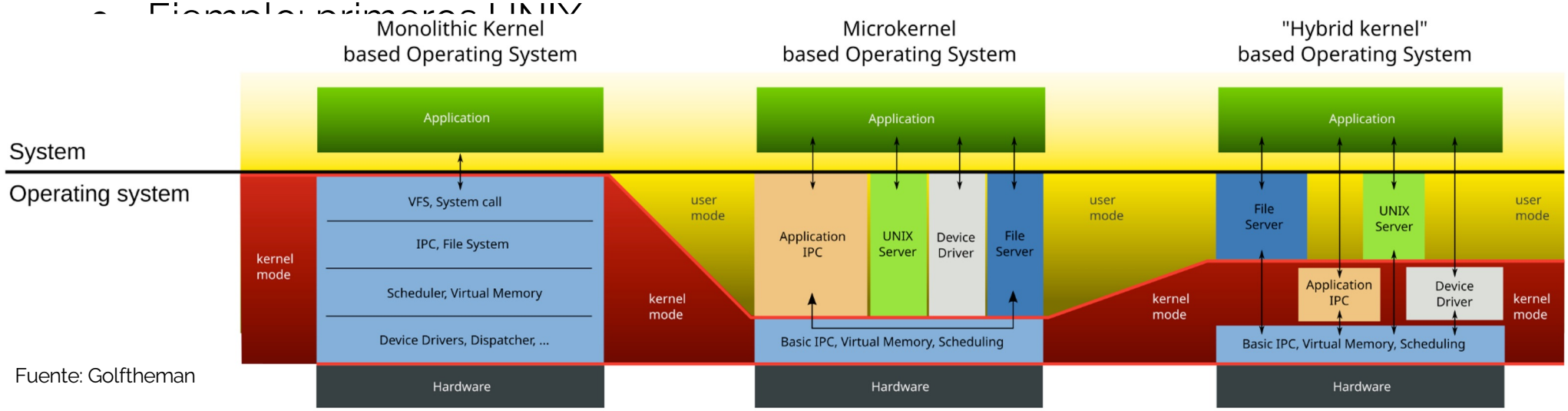
- Apertura, lectura, escritura y cierre de archivos.
- Modificación de atributos (permisos, fechas).
- Navegación por el sistema de archivos.
- Control de acceso y bloqueo de archivos.
- Información del sistema: identificadores de proceso, información de hardware, tiempo del sistema.
- Configuración del entorno: locales, variables de entorno.
- Control del sistema (apagar, reiniciar).



Shmuel Csaba Otto Traian

Modelo monolítico

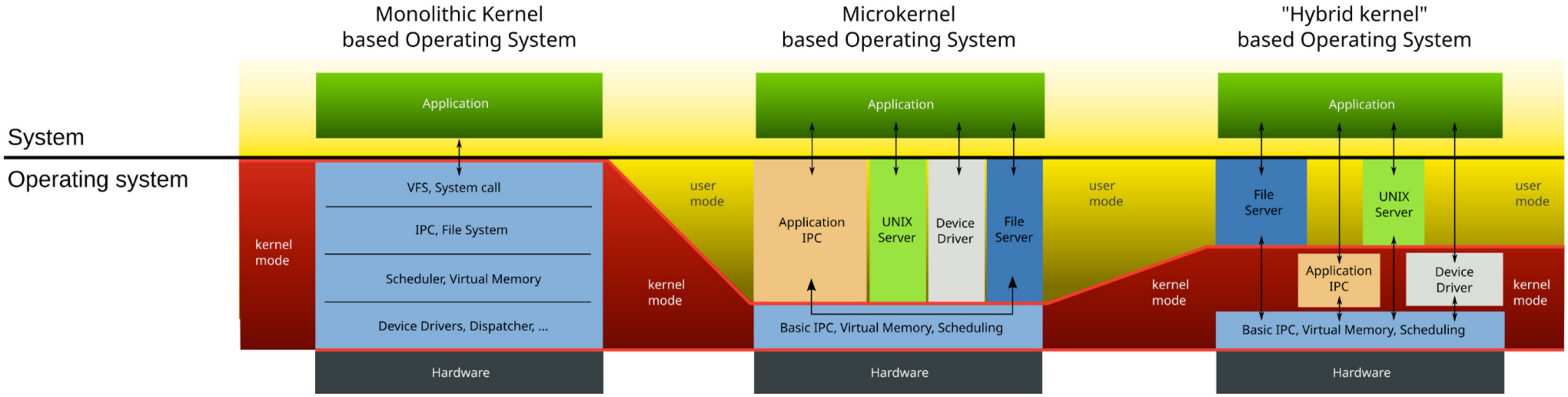
- El sistema operativo se implementa como un solo programa con todas las funcionalidades en el mismo espacio de direcciones.
- Ventajas: alto rendimiento.
- Desventajas: difícil de mantener y depurar.
- Ejemplo: primeros UNIX



Fuente: Golftheman

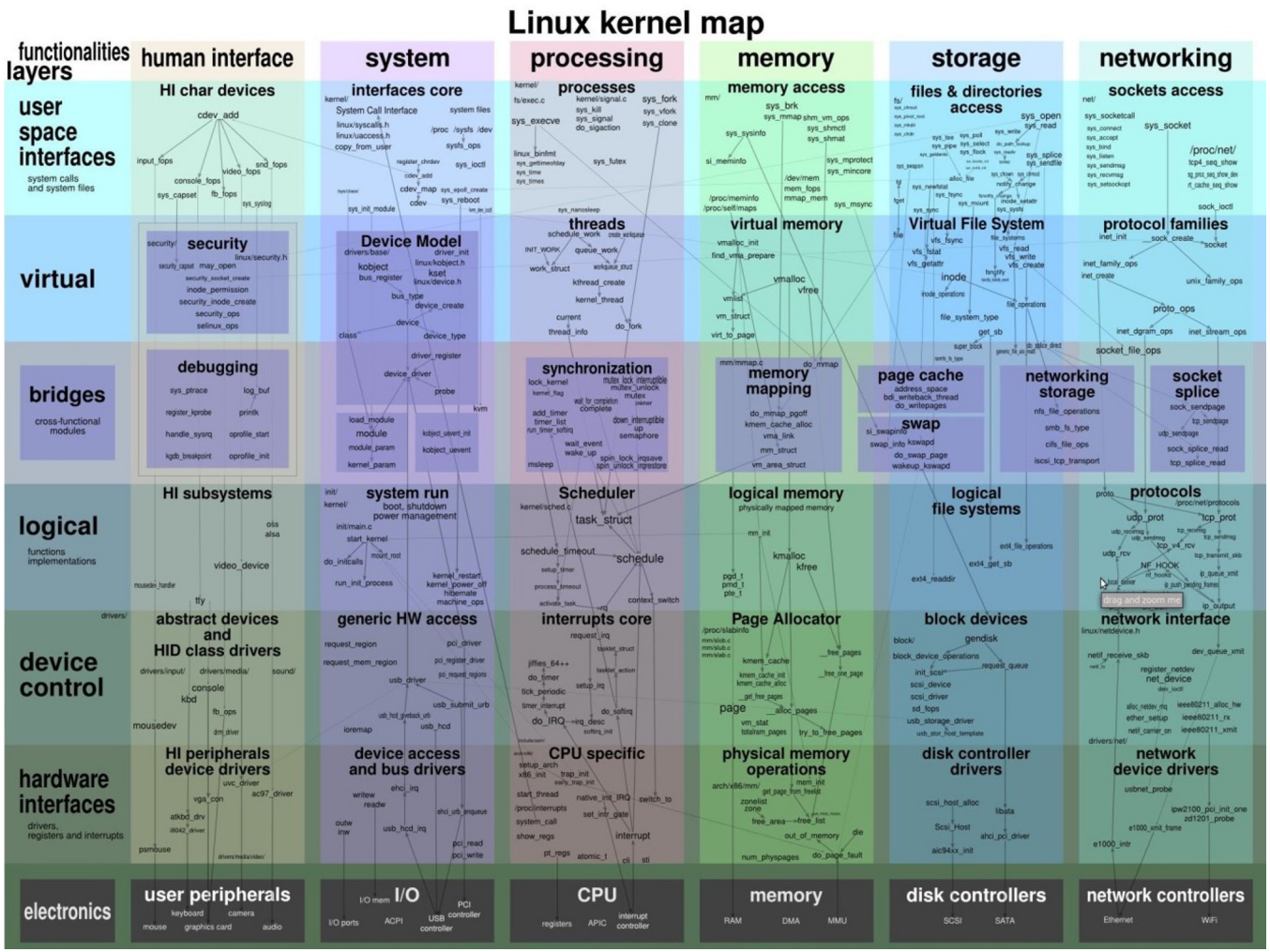
Modelo micronúcleo

- El núcleo contiene solo lo esencial: planificación, comunicación y manejo de interrupciones.
- Otros servicios se ejecutan en espacio de usuario.
- Aumenta la estabilidad y portabilidad.
- Ejemplos: Minix, Mach.



Modelo en capas

- Cada capa implementa un subconjunto de funcionalidades.
- Las capas superiores usan servicios de las inferiores.
- Mejora la depuración, mantenimiento y extensibilidad.



Modelo cliente-servidor

- El núcleo es mínimo y los servicios del SO se ejecutan como procesos separados.
- La comunicación se realiza por mensajes.
- Alta fiabilidad y seguridad.
- Facilita el diseño distribuido.

Comparación de modelos

Modelo	Ventajas	Desventajas
Monolítico	Rápido y eficiente	Difícil de modificar
En capas	Modular, fácil de depurar	Rígido en estructura
Cliente-servidor	Mejor seguridad y mantenimiento	Comunicación costosa
Micronúcleo	Portabilidad, tolerancia a fallos	Menor rendimiento relativo