

# 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:

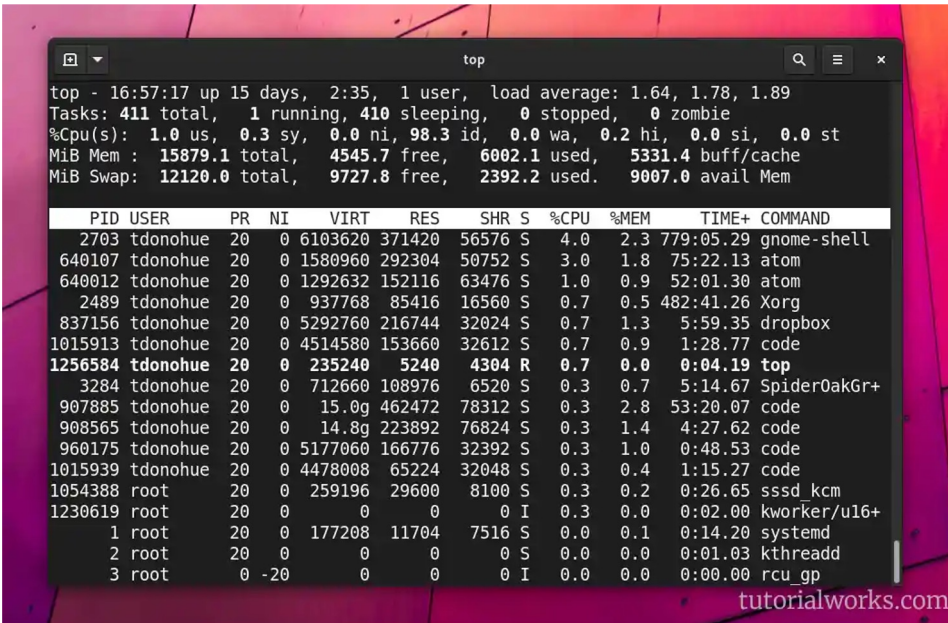
- Línea de comandos (CLI): interfaz textual donde el usuario ejecuta comandos.
- Interfaz gráfica (GUI): permite una interacción más visual y amigable mediante ventanas, menús y dispositivos apuntadores. El SO actúa como puente entre las solicitudes del usuario y los recursos del sistema.



A terminal window with a dark blue background. The prompt is `(base)`. The window title bar shows three tabs: `~ (-zsh)`, `..zab.github.i...`, and `~ (-zsh)`. The terminal content shows `(base)` followed by a tilde `~`, and then `(base)` followed by another `(base)`.



## Servicios del SO (2) - Manejo de procesos



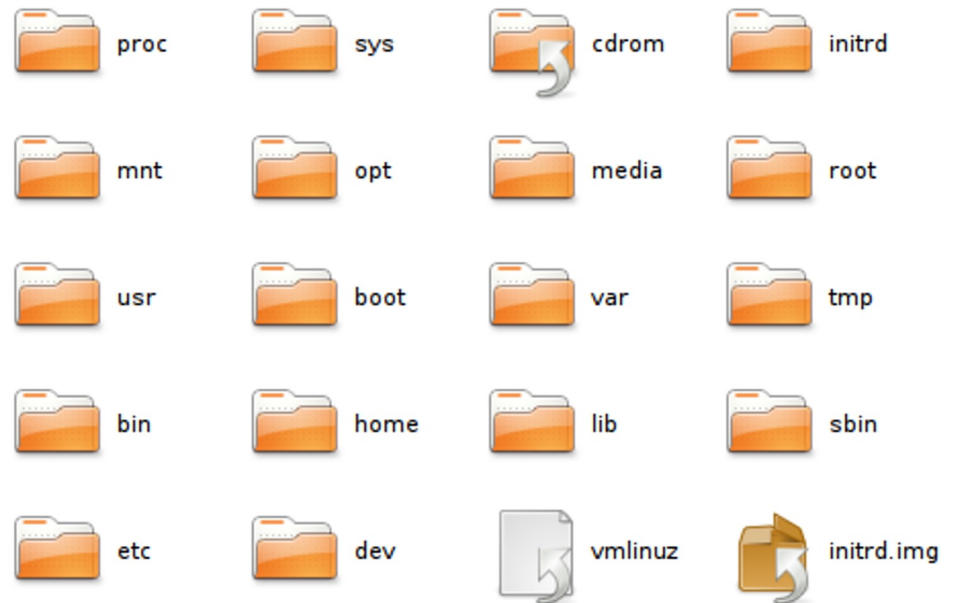
```
top - 16:57:17 up 15 days, 2:35, 1 user, load average: 1.64, 1.78, 1.89
Tasks: 411 total, 1 running, 410 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.0 us, 0.3 sy, 0.0 ni, 98.3 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 15879.1 total, 4545.7 free, 6002.1 used, 5331.4 buff/cache
MiB Swap: 12120.0 total, 9727.8 free, 2392.2 used, 9007.0 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
 2703 tdonohue  20   0 6103620 371420 56576 S   4.0   2.3 779:05.29 gnome-shell
640107 tdonohue  20   0 1580960 292304 50752 S   3.0   1.8 75:22.13 atom
640012 tdonohue  20   0 1292632 152116 63476 S   1.0   0.9 52:01.30 atom
 2489 tdonohue  20   0 937768 85416 16560 S   0.7   0.5 482:41.26 Xorg
837156 tdonohue  20   0 5292760 216744 32024 S   0.7   1.3 5:59.35 dropbox
1015913 tdonohue  20   0 4514580 153660 32612 S   0.7   0.9 1:28.77 code
1256584 tdonohue  20   0 235240 5240 4304 R   0.7   0.0 0:04.19 top
 3284 tdonohue  20   0 712660 108976 6520 S   0.3   0.7 5:14.67 SpiderOakGr+
907885 tdonohue  20   0 15.0g 462472 78312 S   0.3   2.8 53:20.07 code
908565 tdonohue  20   0 14.8g 223892 76824 S   0.3   1.4 4:27.62 code
960175 tdonohue  20   0 5177060 166776 32392 S   0.3   1.0 0:48.53 code
1015939 tdonohue  20   0 4478008 65224 32048 S   0.3   0.4 1:15.27 code
1054388 root      20   0 259196 29600 8100 S   0.3   0.2 0:26.65 sssd_kcm
1230619 root      20   0 0 0 0 I   0.3   0.0 0:02.00 kworker/u16+
 1 root      20   0 177208 11704 7516 S   0.0   0.1 0:14.20 systemd
 2 root      20   0 0 0 0 S   0.0   0.0 0:01.03 kthreadd
 3 root      0 -20 0 0 0 I   0.0   0.0 0:00.00 rcu_gp
```

- 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.



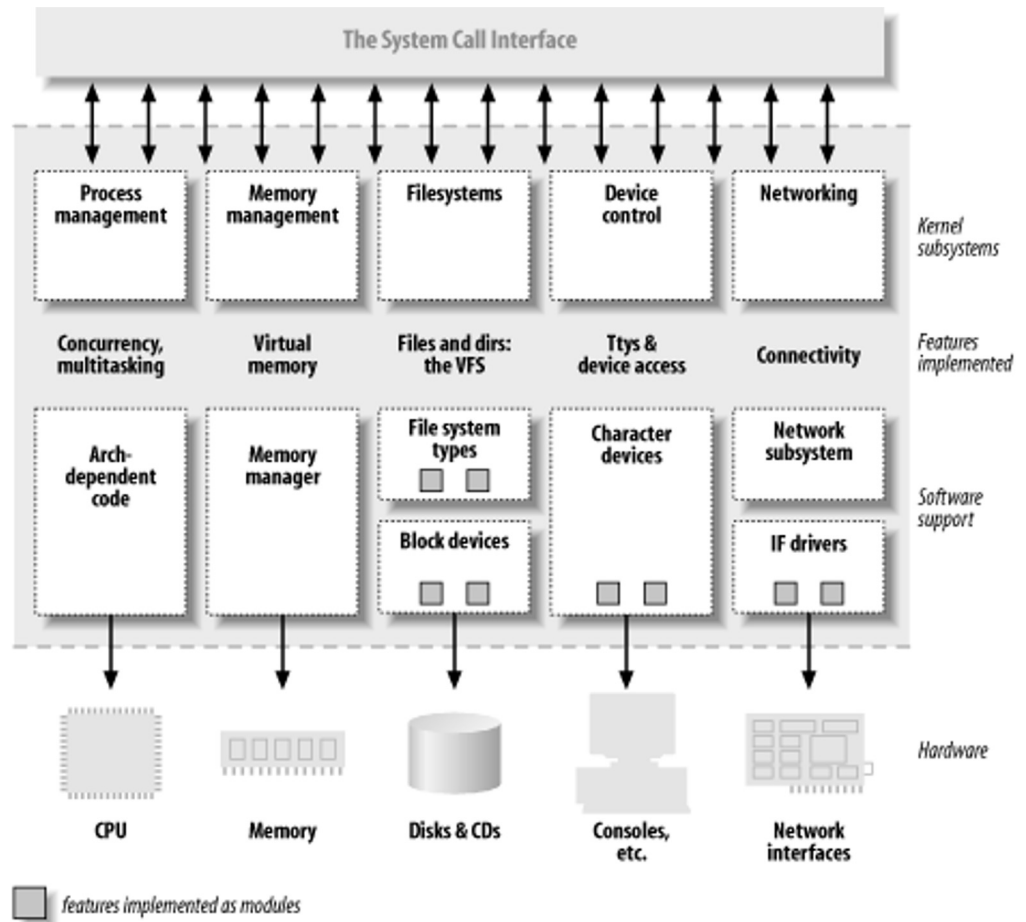
## Servicios del SO (6) - Seguridad

- 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

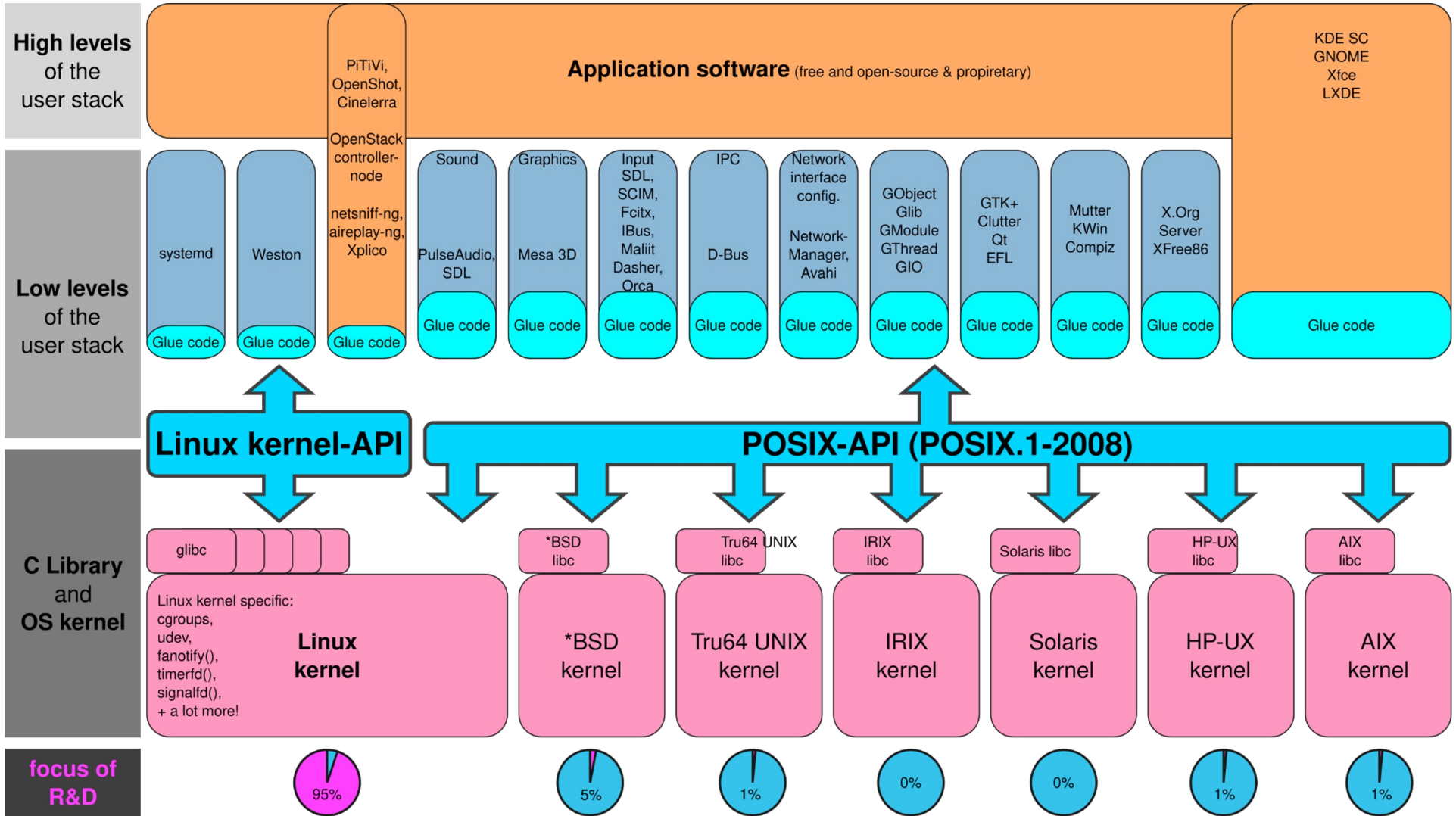
- Las aplicaciones invocan funciones definidas en una API (por ejemplo POSIX, WinAPI).
- Las API traducen estas funciones en llamadas al sistema reales.
- Ejemplo: `open("archivo.txt")` se convierte en una syscall de apertura de archivo.

## 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.

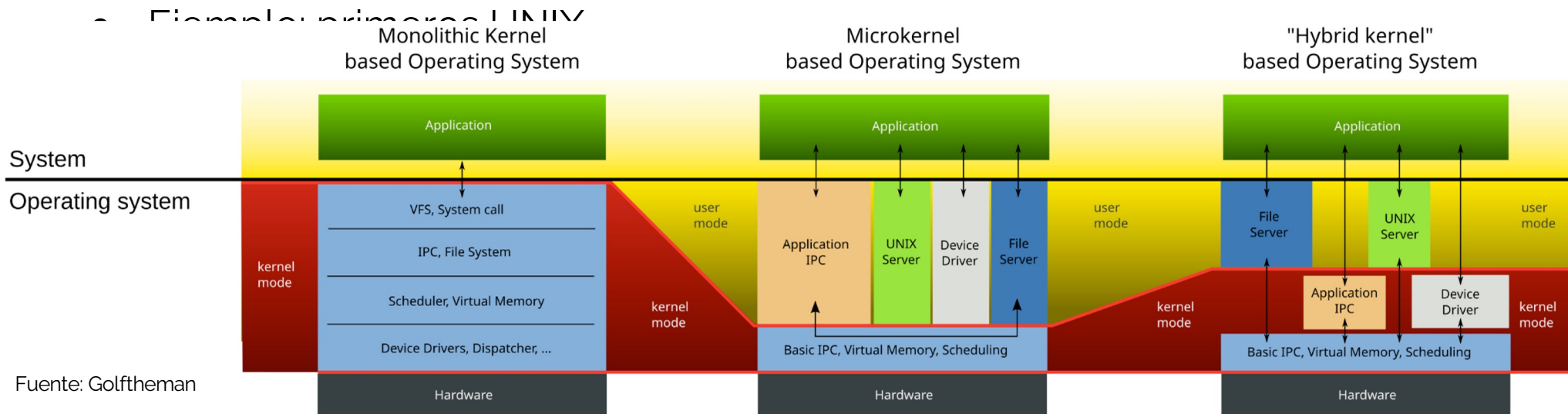
## Tipos de llamadas al sistema (2) - Archivos - Sistema

- 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).



# Modelo monolítico

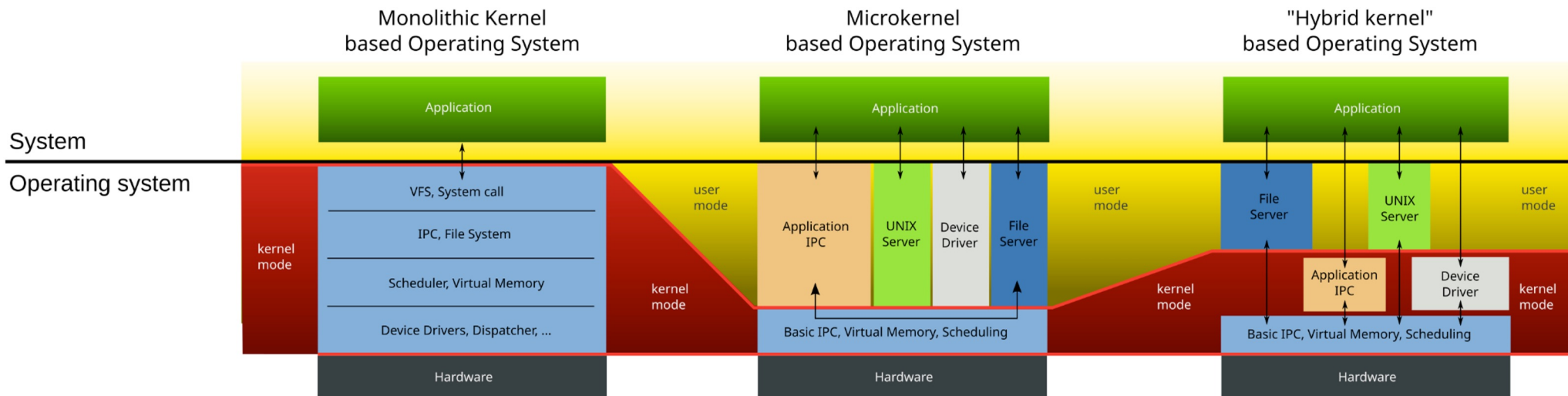
- El sistema operativo se implementa como un solo programa con todas las funcionalidades en el mismo espacio de direcciones.
- Ventajas: alto rendimiento.
- Desventajas: difícil de mantener y depurar.



Fuente: Golftheman

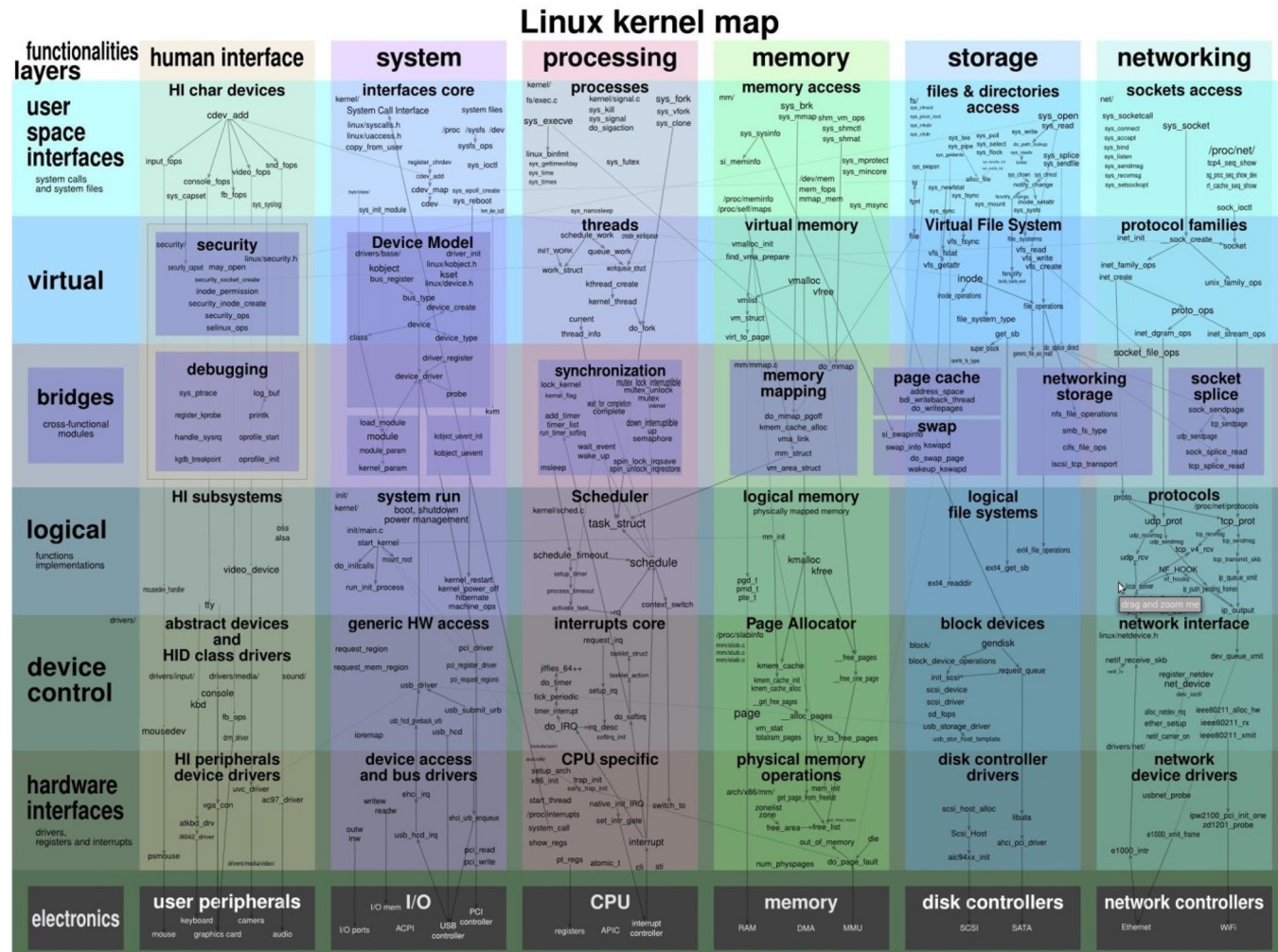
# Modelo micronúcleo

- 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