

Tabla de Contenidos

1. Introducción a los Sistemas Operativos

1.1 Fundamentación de la necesidad de un Sistema Operativo

1.2 Historia de los S.O.

1.3 Familias de Sistemas Operativos

2. Referencias bibliográficas

1.1 Fundamentación de la necesidad de un Sistema Operativo

- ¿Qué es un Sistema Operativo?
- ¿Qué hace un Sistema Operativo
- ¿Algún ejemplo de Sistema Operativo?
 - Para desktops
 - Para smart phones

1.1 Fundamentación de la necesidad de un Sistema Operativo

- ¿Es una Máquina Virtual un SO?
- ¿Es Android o iOS un SO?
- ¿Es una JVM (Java Virtual Machine) un SO?



Fig. 1 : Más tipos de Sistemas Operativos

¿Qué es un SO?

Sistema de Información

Software (programas). Hardware (máquina física y componentes electrónicos).

Sistema Operativo

Un SO es un programa que controla la ejecución de programas de usuario y actúa como intermediario entre los usuarios y el **hardware** de la computadora.

- Es una representación **abstracta** de los recursos que pueden ser utilizados y requeridos por las aplicaciones (Procesador, memoria, I/O (disk, network))
- Hacer que el Sistema de Información sea fácil de utilizar.
- Usar el hardware de la computadora de una manera más eficiente

¿Qué es un SO?

Objetivos

- Hacer que el uso del Sistema de Información se más fácil
- Hacer que el uso del Sistema de Información sea más eficiente
- Hacer que el uso del Sistema de Información sea más seguro

¿Qué es un SO?

El Sistema Operativo es una capa de Software entre las aplicaciones y hardware de la computadora

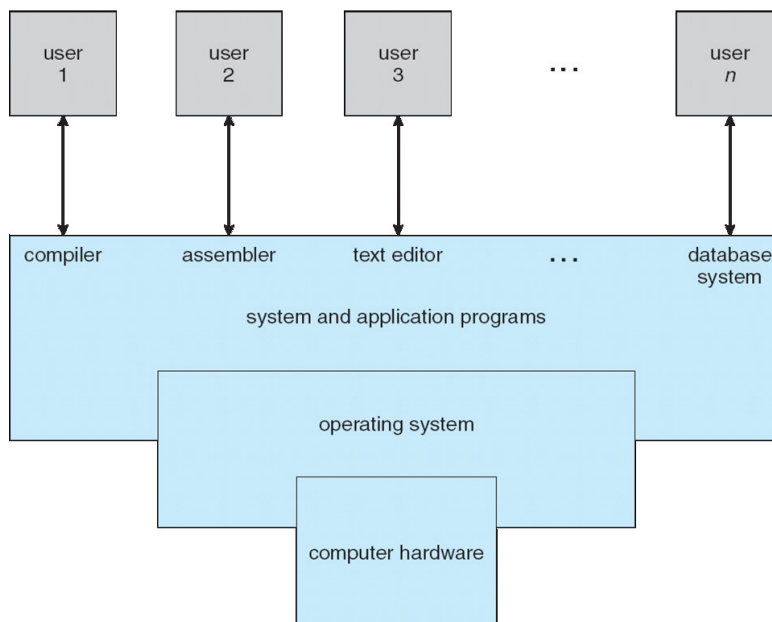


Fig. 2 : Componentes de un SO

Componentes de un Sistema Operativo

- Hardware
 - Componentes Básicos: processor (CPU), memory, I/O y dispositivos (devices)
- Sistema Operativo
 - Controla y coordina el uso del Hardware en medio de múltiples programas ejecutándose en un computador (PC).
- Aplicaciones
 - Solucionan problemas específicos de usuario: compiladores, sistemas de bases de datos, aplicaciones de gestión
- Usuario
 - Personas, u otras aplicaciones de usuario (procesos de inter-comunicación, sistemas distribuidos).

Rol de un Sistema Operativo

- Proveedor de Servicios
 - Conjunto de servicios para los usuarios del sistema
- Reserva de Recursos
 - Explota los recursos HW de uno o más procesadores y los reserva para los programas de usuarios
- Control de Programas
 - Controla la ejecución de programas y operaciones de Dispositivos de E/S (interrumpiéndolos para enviar/recibir datos vía E/S o para reservar recursos hardware a otros usuarios.
- Protección y Seguridad
 - Proteger la ejecución de múltiples programas
 - Securitizar el acceso del usuario a los datos y definir la propiedad de los archivos/directorios y procesos

Organización de un Sistema Informático

- Una o más CPUs,
- Controladores de Dispositivo que se conectan a través de **buses** y que acceden a memoria compartida
- Memoria

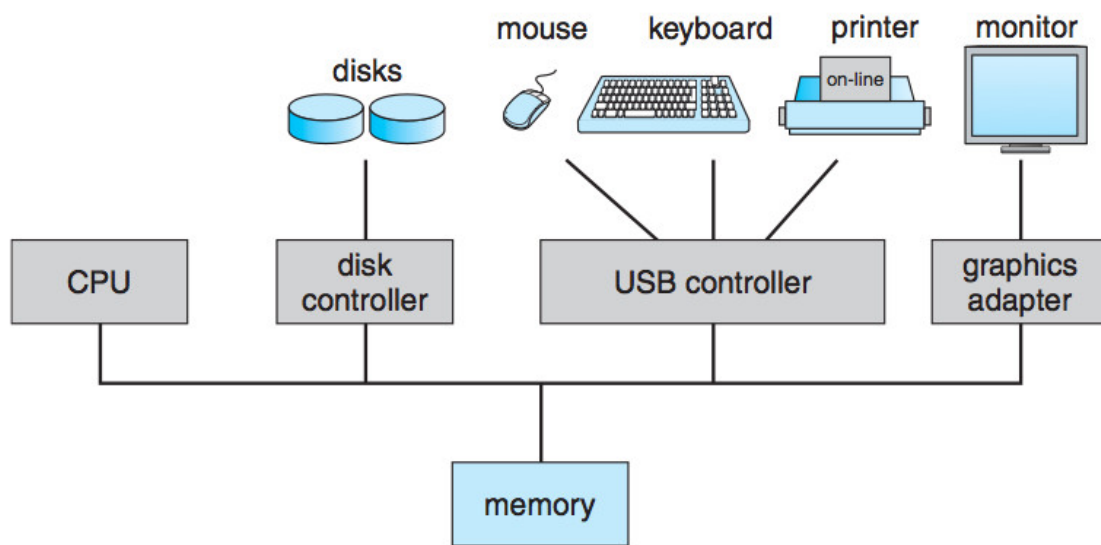


Fig. 3 : Organización

Modos de Operación

El Sistema Operativo se puede ejecutar en modo Dual para protegerse asimismo de otros componentes (o usuarios) del sistema

- Modo usuario y Modo Kernel (o privilegiado)
- El bit de modo:
 - Para distinguir cuándo el sistema está ejecutando código de usuario o código del kernel
 - Algunas instrucciones están designadas como **privilegiadas** (p.e. E/S)

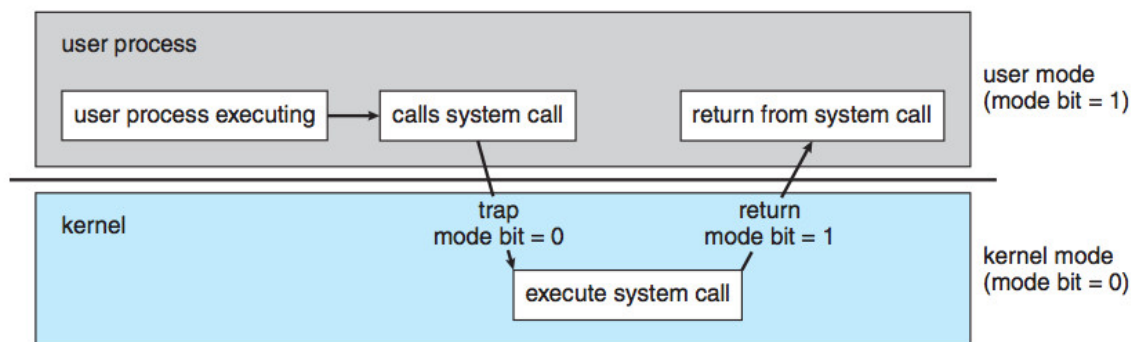


Fig. 5 : Modo Kernel y Usuario

Protección y Seguridad

- **Protección**: cualquier mecanismo para controlar el acceso de los procesos o usuario a los recursos definidos por el SO.
- **Seguridad**: defensa del SO ante ataques tanto internos como externos (DoS, worms, virus, suplantación de identidad)
- El SO distingue de entre varios tipos de usuarios que determinan qué pueden realizar:
 - Las identidades de usuario (user IDs) incluyen nombre y número asociado (unívoco)
 - User ID se asocia con todos los ficheros y procesos de este usuario
 - Group ID permite que un conjunto de usuarios sean gestionados con un identificativo de proceso asociado
 - Aumento de privilegios: permite al usuario cambiar el ID efectivo con más derechos

Servicios de los SO

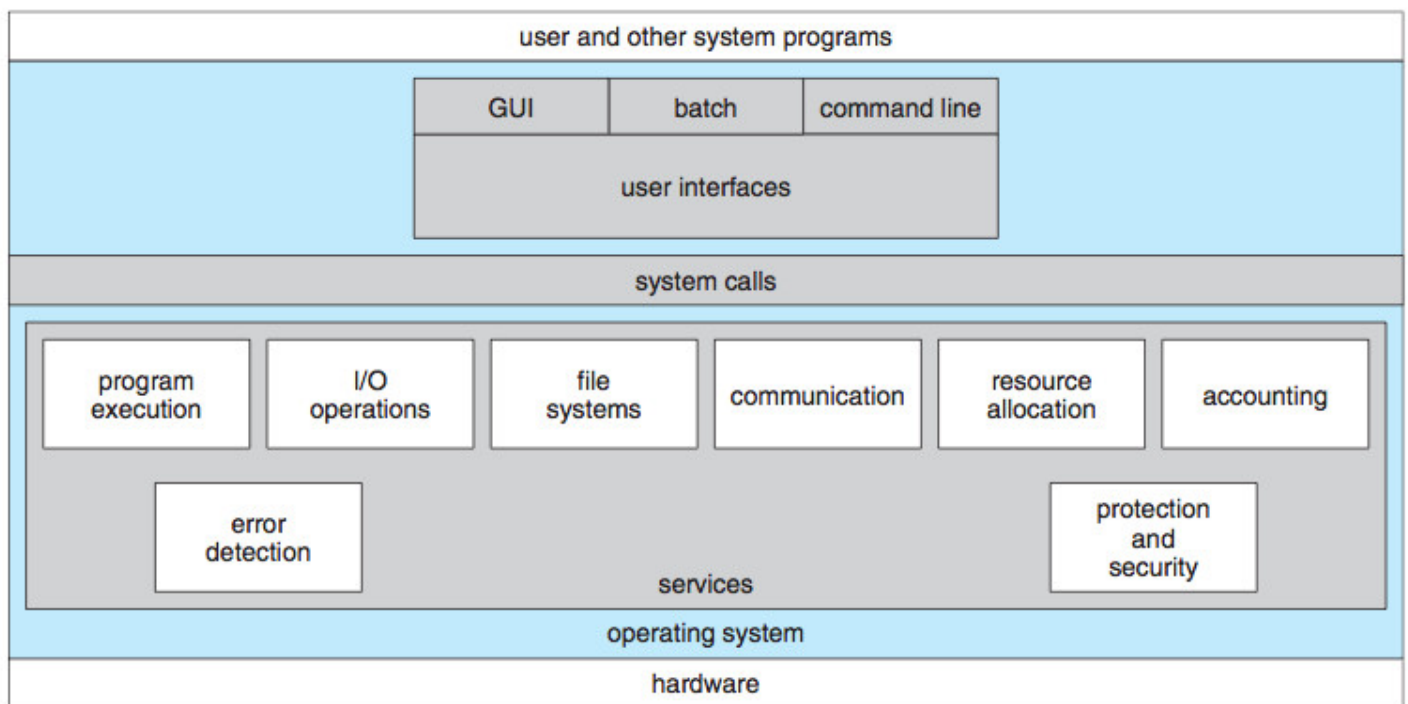


Fig. 6 : Servicios del SO

Servicios de los SO

Los SO proveen un entorno para la ejecución de programas y servicios para las aplicaciones de usuario. Este conjunto de servicios del SO nos dan funciones muy útiles para el usuario:

- **Interfaz de Usuario (UI):** Puede ser Interfaz de Línea de Comandos (CLI) o Interfaz Gráfica de Usuario (GUI).
- **Ejecución de Programas:** El sistema debe poder cargar un programa en memoria y ejecutarlo (si da error, debe informar).
- **Operaciones E/S:** Un programa en ejecución requiere E/S cuando utilizar un dispositivo o un fichero.
- **Sistema de Ficheros:** Los programas necesitan leer y escribir en ficheros y directorios, además de crear y borrarlos, listarlos, asignar permisos.
- **Comunicaciones:** Los procesos intercambian información en la misma computadora o con otras a través de la red (mediante memoria compartida o paso de mensajes).
- **Detección de errores:** El OS debe informar de errores constantemente (ocasionados por la CPU, memoria, dispositivos, programas)

Servicios de los SO

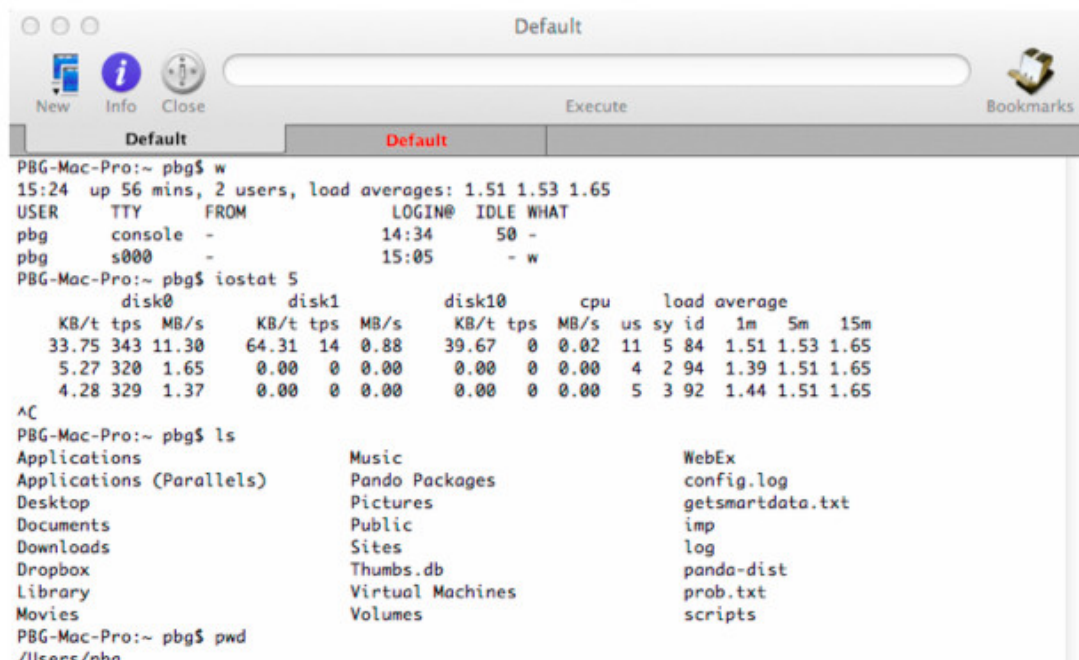
Otros servicios que aseguran la eficiencia de operaciones del sistema mediante la comparación de recursos:

- **Reserva de Recursos:** Múltiples usuarios o procesos ejecutándose concurrentemente (recursos como CPU, memoria, almacenamiento de ficheros, E/S).
- **Contabilidad:** Para mantener un registro de los usuarios que utilizan los recursos
- **Protección y seguridad:** Los propietarios de la información guardada en un computador de red o multiusuario controlan el uso de esa información, y los procesos concurrentes no deberían interferirse asimismos.
 - Protección: asegura que el acceso a todos los recursos de sistema sea controlado
 - Seguridad: del sistema frente a eventos externos requiere autenticación de usuario, además de intentos de acceso mediante dispositivos E/S.

CLI

El CLI (Command Line Interface) o Intérprete de Comandos permite introducir comandos directamente:

- Implementado en el kernel, a veces en programas de sistema
- También llamado **shell**
- El usuario introduce un comando y el sistema lo ejecuta



```
Default
New Info Close Execute Bookmarks

PBG-Mac-Pro:~ pbg$ w
15:24 up 56 mins, 2 users, load averages: 1.51 1.53 1.65
USER      TTY      FROM          LOGIN@  IDLE WHAT
pbg       console -            14:34    50 -
pbg       s000    -            15:05    - w
PBG-Mac-Pro:~ pbg$ iostat 5
            disk0      disk1      disk10      cpu      load average
      KB/t tps MB/s    KB/t tps MB/s    KB/t tps MB/s    us sy id  1m  5m 15m
      33.75 343 11.30    64.31 14  0.88    39.67  0  0.02    11  5 84  1.51 1.53 1.65
      5.27 320  1.65      0.00  0  0.00      0.00  0  0.00     4  2 94  1.39 1.51 1.65
      4.28 329  1.37      0.00  0  0.00      0.00  0  0.00     5  3 92  1.44 1.51 1.65
^C
PBG-Mac-Pro:~ pbg$ ls
Applications                               Music
Applications (Parallels)                   Pando Packages
Desktop                                    Pictures
Documents                                  Public
Downloads                                  Sites
Dropbox                                   Thumbs.db
Library                                   Virtual Machines
Movies                                   Volumes
PBG-Mac-Pro:~ pbg$ pwd
/Users/pbg
WebEx
config.log
getsmartdata.txt
imp
log
panda-dist
prob.txt
scripts
```

GUI

Es la interfaz *metáfora* de un Escritorio:

- Ratón, teclado, monitor, iconos (ficheros, programas, acciones, etc.)
- Acciones del Mouse pueden dar más información (doble clic, contextual)
- Inventado en Xerox PARC

Muchos SO incluyen tanto CLI como GUI

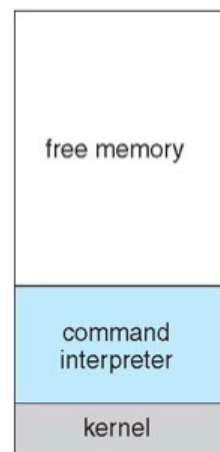
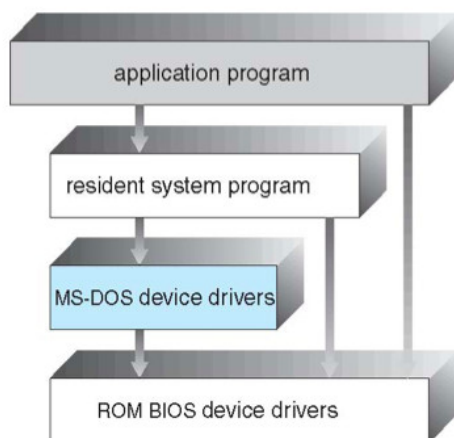
- Microsoft Windows tiene GUI y CLI (command *shell*)
- Apple Mac OS X tiene un GUI *Aqua* y un CLI basado en UNIX
- Unix y Linux tienen CLI y GUI opcionales (CDE, KDE, GNOME)

1.3 Familias de Sistemas Operativos

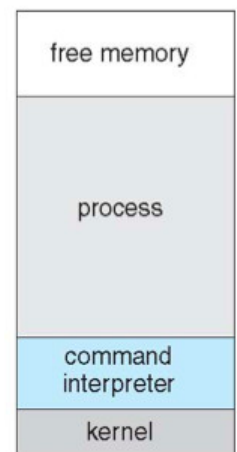
- Sistemas Operativos de Propósito General
- Hay varias formas de clasificarlos o estructurarlos:
 - Monolíticos
 - Por Capas
 - Microkernel
 - Híbridos

Monolíticos: Estructura de MS-DOS

- Hecho para proveer la mayor funcionalidad en el menor espacio de tiempo
 - No se divide en módulos: sus interfaces y niveles de funcionalidad no están bien separados.
 - Tareas simples, shell invocado cuando el sistema es iniciado (no se crean procesos).
 - El programa se carga en memoria sobrescribiendo todo menos el kernel.



(a)



(b)

Monolíticos: Estructura de Unix

Unix estaba limitado por la funcionalidad del HW. Consistía en dos partes: kernel y programas del sistema

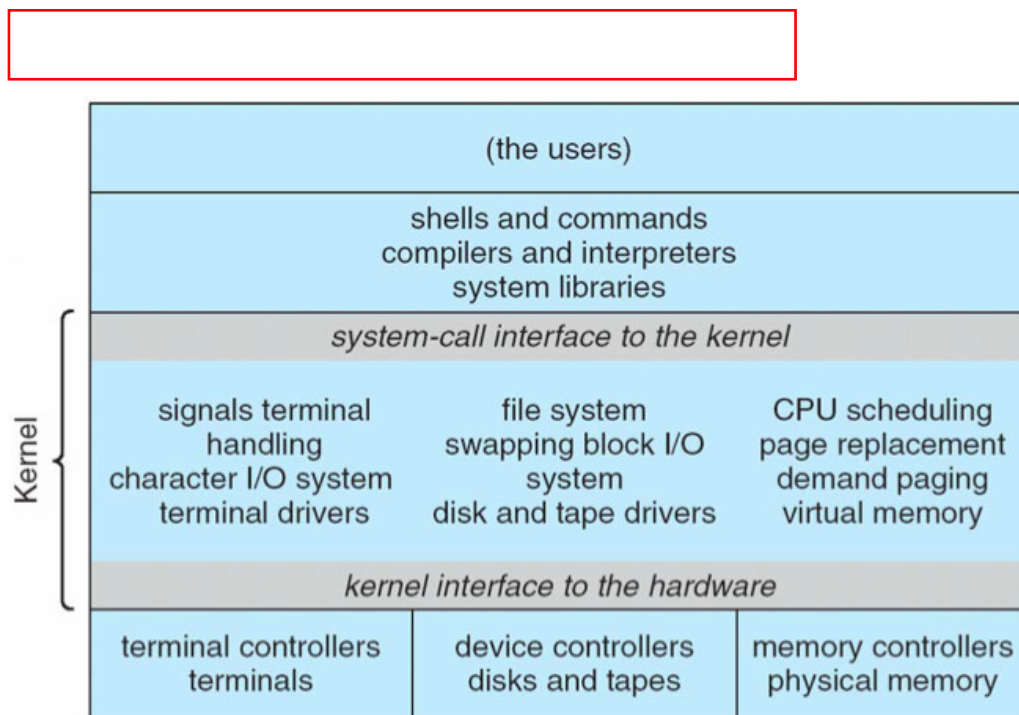


Fig. 14 : UNIX

Por Capas

- Fue MULTICS quien lo perfeccionó a partir de THE.
- Los Unix modernos (y Linux) están basados en esta estructura.
- Numero de capas dispuestas de dentro a afuera, cada una de ellas provee de servicios a la próxima capa
- La capa más interna es el HW mientras que la más externa es la interfaz de usuario
- Tiene la ventaja de que puede ser depurada independientemente.
- Problema en decidir en qué orden se deben disponer y en la comunicación con capas superiores.

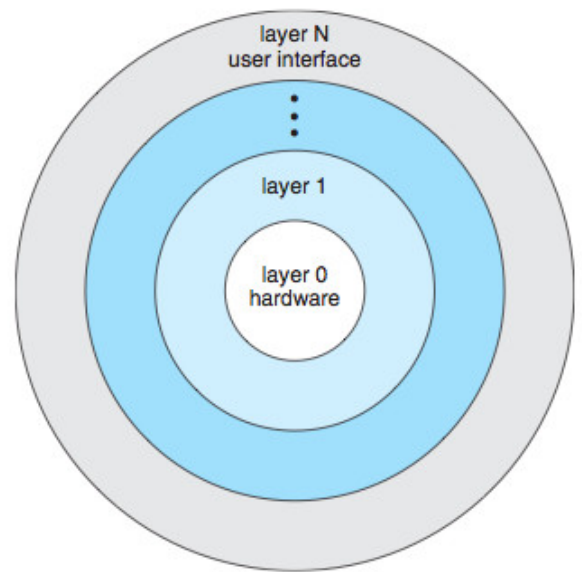


Fig. 16 : SO por Capas

Microkernel: Windows NT

- La idea básica detrás es eliminar todos los servicios no esenciales del kernel e implementarlos como aplicaciones de sistema: se consigue un kernel más pequeño y eficiente.
- Muchos microkernels proveen una gestión básica de memoria y procesos, además de paso de mensajes entre servicios.
- La seguridad y protección están salvaguardados por ser ejecutados en modo usuario, no en modo kernel.
- *Mach* fue el primer SO microkernel y Mac OS X tiene muchos componentes del mismo.
- *Windows NT* fue originariamente microkernel, debido a los problemas de rendimiento de Windows 95. NT mejoró el rendimiento, aunque Windows XP fue más monolítico.

Microkernel: Windows NT

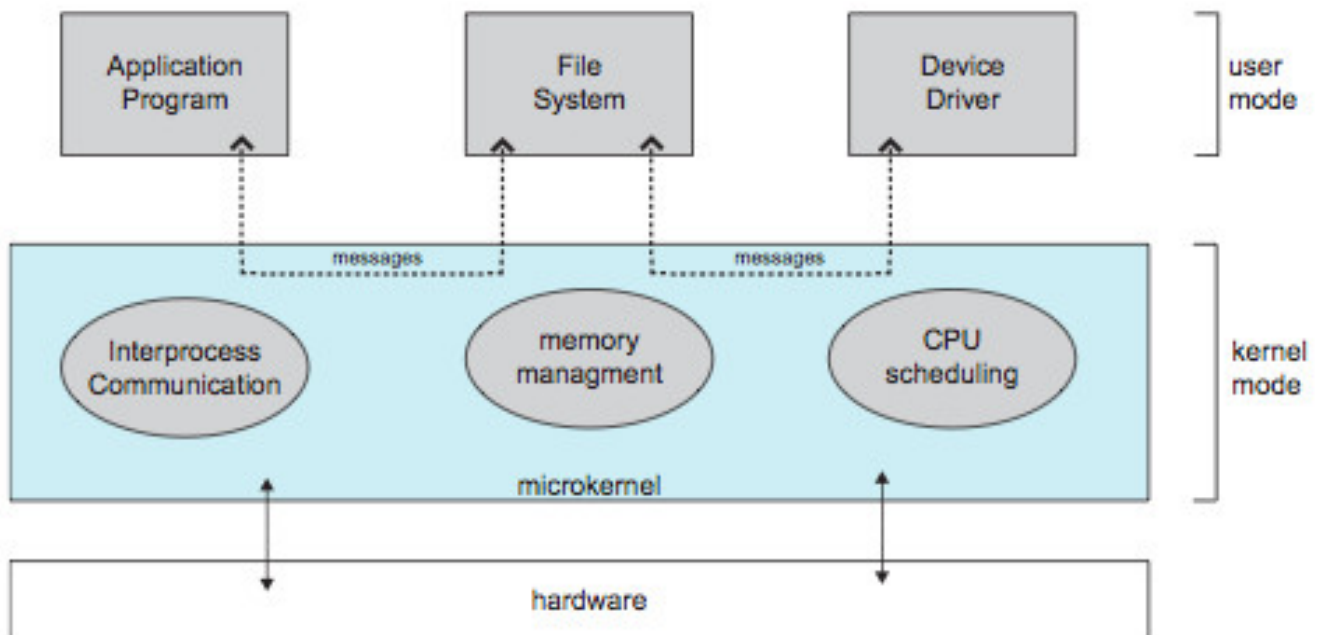
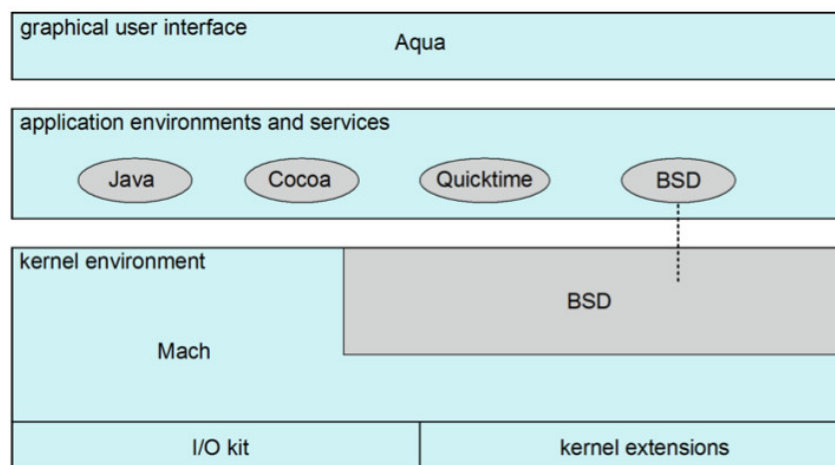


Fig. 17 : Microkernel Windows NT

SO Híbridos: Mac OS X

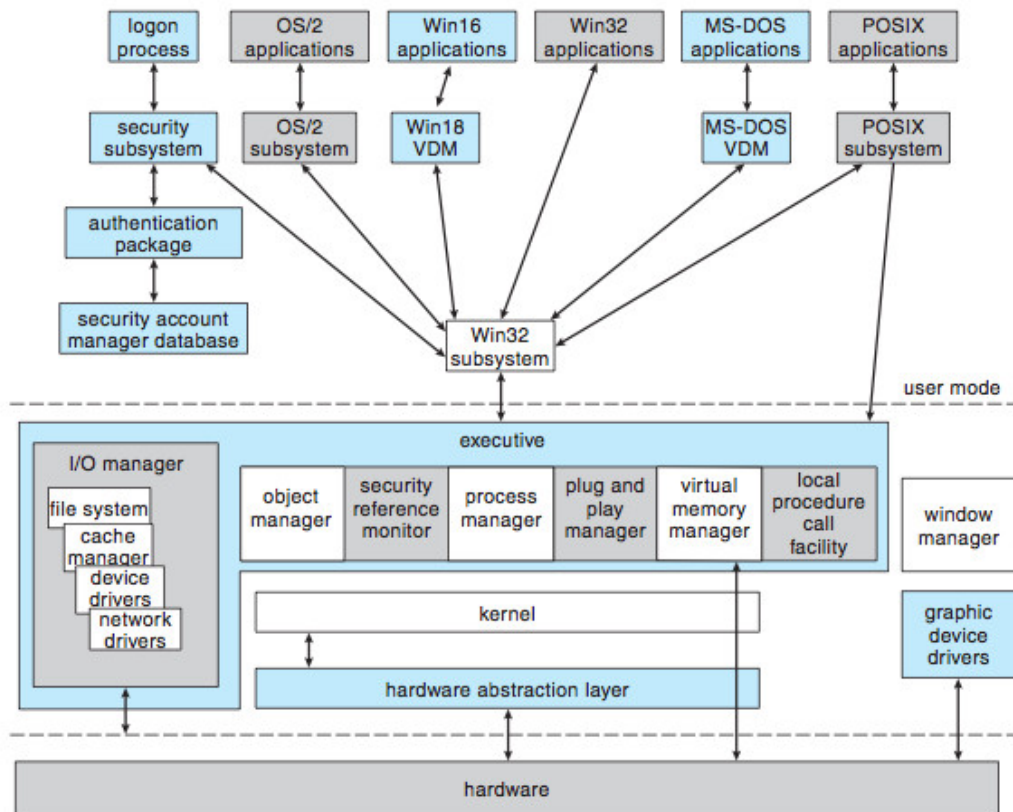
Muchos de los SO de hoy no están adheridos a una única estructura, tomando una mezcla de varias arquitecturas.

- La arquitectura de Mac OS X toma aspectos del microkernel de Mach para servicios de gestión de sistema básicos y el kernel de BSD para servicios adicionales.
- Los servicios de aplicación son dinámicamente cargados por los módulos (kernel extensiones) y proveen el resto de funcionalidad del SO.



SO Híbridos: Windows 7

Windows llega a ser monolítico, microkernel y para diferentes subsistemas tiene lo que se llaman *personalities*



SO Híbridos: iOS

Los SO de Apple de esta gama son para iPhone y iPad

- Estructurado como en Mac OS X con funcionalidad añadida, basado en su kernel
- No ejecutan aplicaciones OSX de forma nativa (tienen diferente CPU: ARM vs Intel)
- Cocoa Touch: API en Objective-C para desarrollar apps
- Media services: capa para gráficos, audio y video
- Core services: provee cloud computing, bases de datos



SO Híbridos: Android

Desarrollado por Open Handset Alliance (y comprado por Google), es Open Source

- Similar a iOS
- Basado en el kernel de Linux pero modificado:
 - Provee servicios, memoria, gestión de dispositivos
 - Gestión de Energía
- Se ejecuta en un entorno de ejecución que incluye un conjunto de librerías y una máquina virtual (Dalvik)
- Las apps están desarrollada en Java en una API de Android distinta
- Las Librerías incluyen frameworks para navegadores (webkit), Sqlite, multimedia, NFC

SO Híbridos: Android

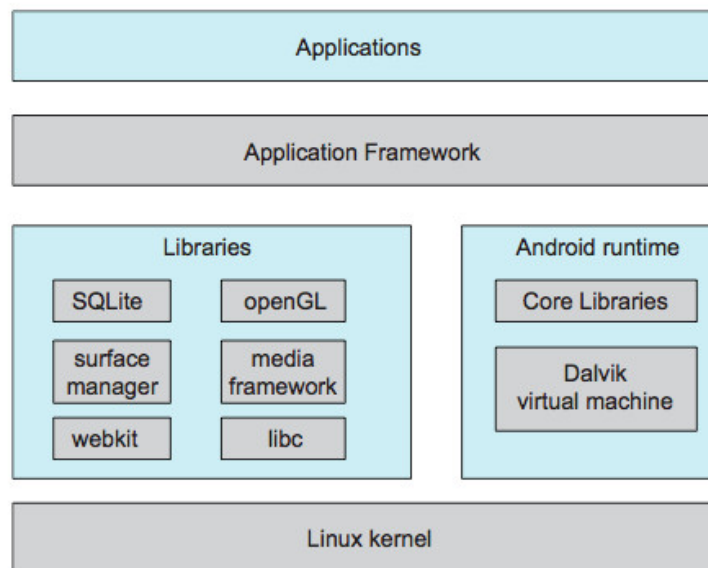


Fig. 21 : Google Android

Comparación de híbrido-microkernel: <http://upload.wikimedia.org/wikipedia/commons/d/d0/OS-structure2.svg>

Referencias bibliográficas I



"FUNDAMENTOS DE SISTEMAS
OPERATIVOS" (Capítulo 1).
Gunnar Wolf y otros
CREATIVE COMMONS
(Disponible en moodle)



[3] [1] [2]