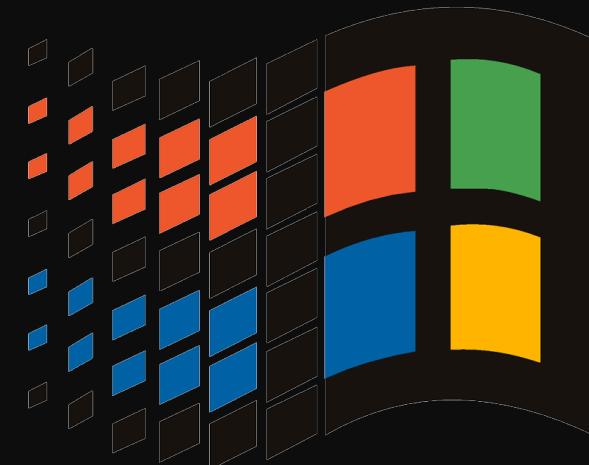


SISTEMAS OPERATIVOS

AYUDANTÍA I



AYUDANTE: YERKO ORTIZ



EL SISTEMA OPERATIVO - SLIDE I

- Hace muchos años atrás, una definición precisa y acotada de sistema operativo hubiese sido: el software que controla a el hardware.
- El problema de esta definición es que es incompleta (no obsoleta) para hoy en día, debido a la constante evolución que han sufrido los sistemas operativos y el hardware.

EL SISTEMA OPERATIVO - SLIDE II

- Imagine lo siguiente, usted ha programado toda su vida en lenguaje assembly y no conoce ningún otro lenguaje.
- Assembly es un lenguaje que permite controlar el hardware de la maquina mediante el conjunto de instrucciones de una arquitectura. Von Neumann (?)
- Continue imaginando, a usted le dan ganas de implementar su videojuego favorito en lenguaje assembly. ¿Cuánto tiempo cree demorar?



EL SISTEMA OPERATIVO - SLIDE III

- Para el videojuego la maquina tendrá muchos programas en ejecución, buffers para lectura de input, el motor gráfico, programas que procesen el input, habrá que gestionar el uso de la memoria para los programas, habrá que intercomunicar los programas entre si, esto suena a una tarea imposible ...

C BUGUEA

El futuro es hoy,



¿Oíste viejo?.

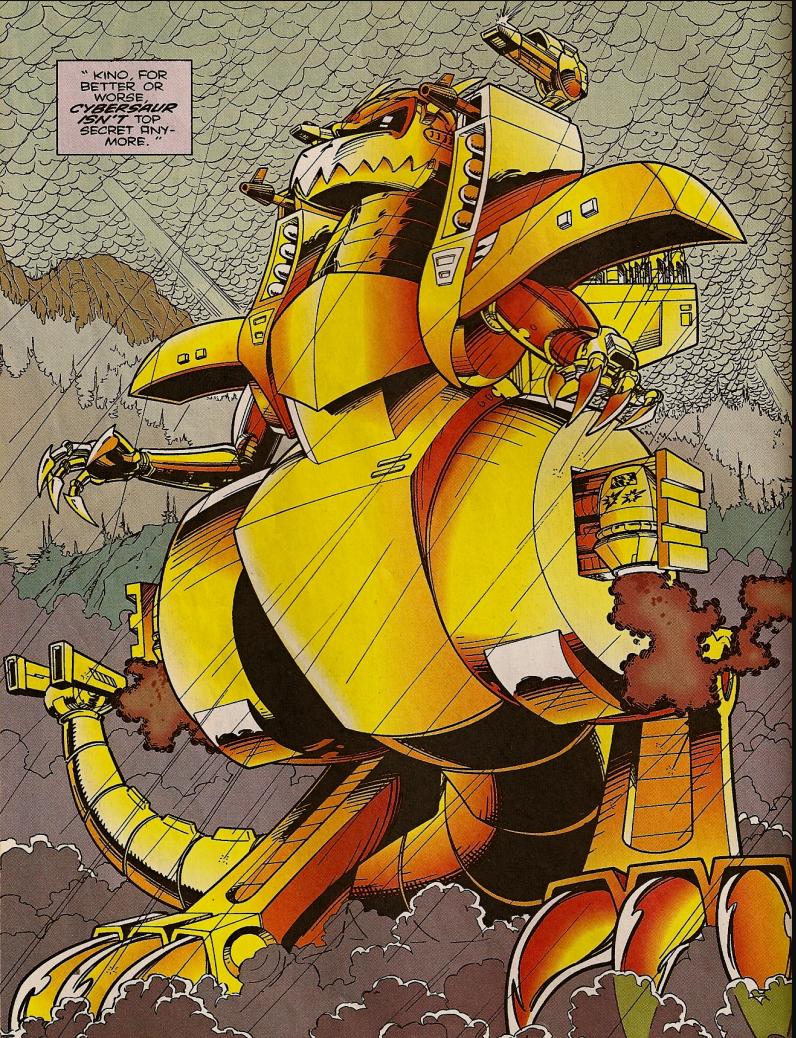
Por suerte los sistemas operativos modernos han llegado al punto de que para usar una computadora no es necesario pensar en las operaciones de hardware que realiza internamente!!!



"Good abstractions turn a nearly impossible task into two manageable ones. The first is defining and implementing the abstraction. The second is using these abstractions to solve the problem at hand". -
(Andrew Tanenbaum, Modern Operating Systems).

Notese que este caballero creó su propio sistema operativo(MINIX) con el objetivo de enseñar sistemas operativos a sus alumnos.

EL SISTEMA OPERATIVO - SLIDE IV



- Finalmente es posible decir que la función del sistema operativo es proveer un conjunto de abstracciones a los programas de usuario, como así mismo administrar los recursos de hardware que se utilizan en la ejecución de programas.
- Es gracias a esas abstracciones que el usuario no ha de entender a la maquina en su totalidad. Esas abstracciones son el objeto de estudio de este curso.

MULTIPROGRAMMING - SLIDE I

- Dejemos de lado la existencia de procesadores multicore, superescalares y otros casos. Por ahora pensemos que solo existen procesadores de un solo núcleo, que ejecutan una instrucción por ciclo de reloj.
- Entonces si solo ejecutan una instrucción por ciclo de reloj, ¿cómo es posible jugar videojuegos?, ¿cómo es posible escuchar música mientras escribo en un editor de texto?, ¿qué es lo que sucede detrás para que eso sea posible con una CPU, que solo ejecuta una instrucción por ciclo de reloj?

MULTIPROGRAMMING - SLIDE II

- Multiprogramming es el concepto que abarca a esas interrogantes, es la capacidad de la maquina de ejecutar distintos programas al mismo tiempo, administrando los recursos de hardware que la ejecución de esos programas requieren.
- Este es el concepto detras cuando realizamos diversas tareas en nuestros dispositivos, sin tener la menor conciencia de que se ejecutan a un ciclo de reloj (por el momento los procesadores multicore y superescalares no existen, ya llegaremos ahí).

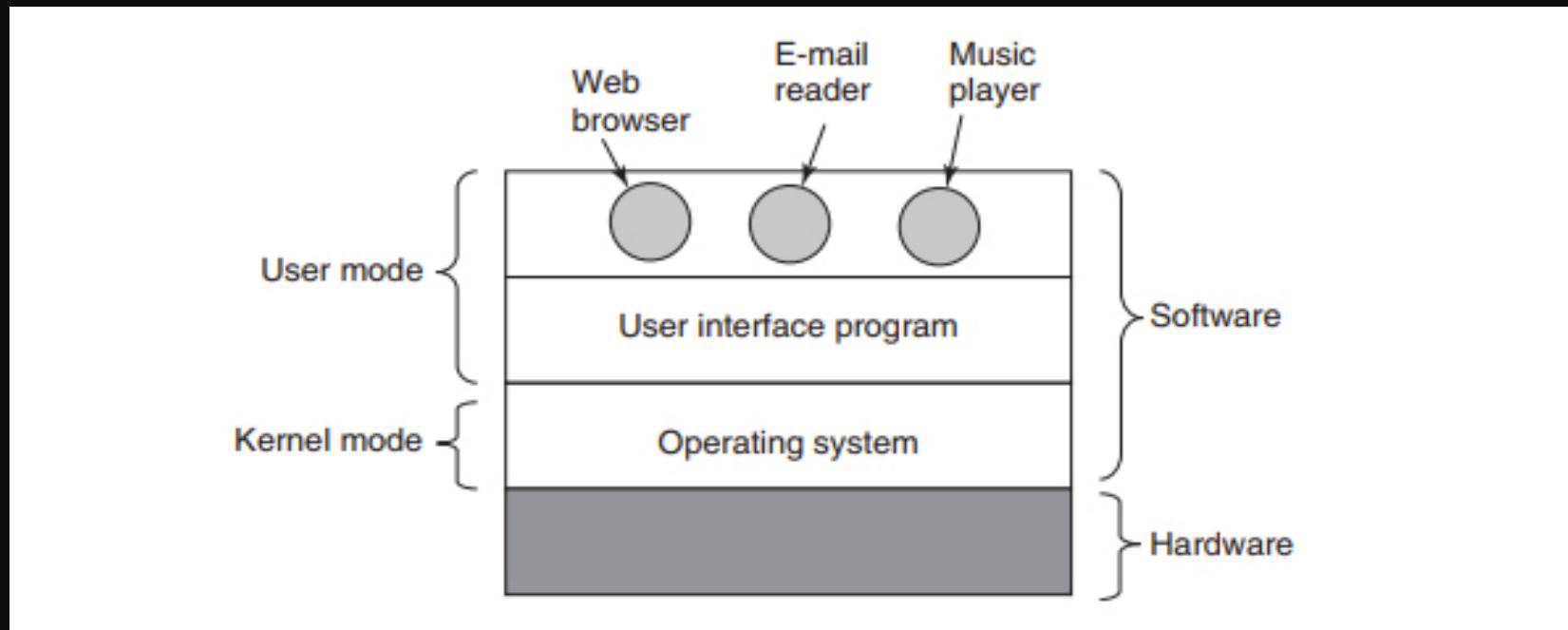
KERNEL

- El kernel de un sistema operativo es su nucleo, si existen errores a nivel de sistema operativo o hardware las primitivas para lidiar con esos errores están en el kernel.
- Aquí están los programas que administran el hardware y deciden como ejecutar los programas.



MODO USUARIO

- Aquí está el espectáculo que los usuarios pueden disfrutar, interpretes de líneas de comandos e interfaces gráficas.
- Aquí están los programas de aplicación que usamos a diario, spotify, lolcito, starcraft, vim, etc.



SYSCALLS

- Las syscalls proveen un conjunto de servicios ofrecidos por el sistema operativo.
- La mayoria está escrita en lenguaje C, aunque algunas operaciones de bajo nivel están escritas en lenguaje assembly.
- Un ejemplo es la syscall fork() que veremos en mayor detalle en el proximo capitulo en este mismo canal, pero que a priori crea un proceso clon de un proceso existente.

SYSCALLS

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount, so no umount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

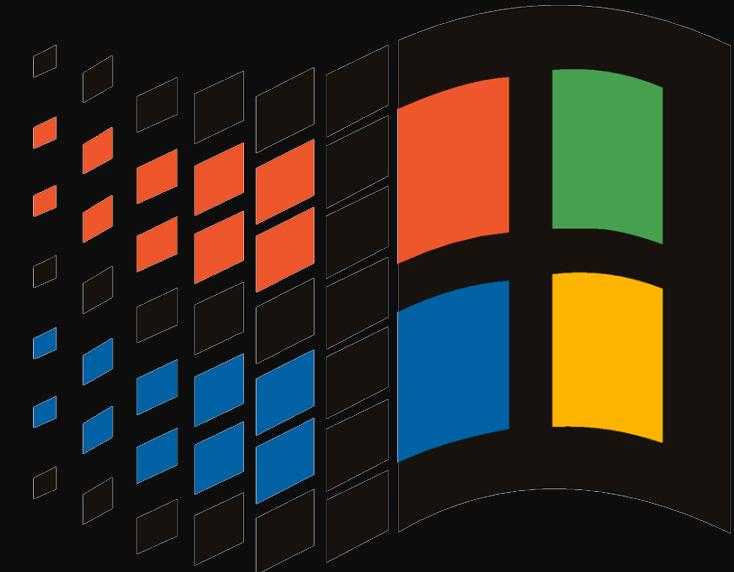
ALGUNOS COMANDOS COOL

- ls, l, cd, echo, pwb, pbcopy, cat, grep, mkdir, chmod, mv.
- top, ps, pstree, pidof, pgrep, kill, pkill, |
- Ejemplo: pwd|pbcopy
- pwd entrega de output el PATH del directorio en que se encuentra actualmente, | dará ese output al comando pbcopy que es el comando que copia, en otras palabras pwd|pbcopy copia en el portapapeles el PATH del directorio actual.

LECTURAS RECOMENDADAS

- Si le gusta la historia, en el capítulo 1, sección 2 del libro de Andrew Tanenbaum, *Modern Operating Systems*, cuentan la historia desde el primer computador (la maquina de Babbage), la primera programadora (Ada Lovelace) , hasta llegar a un joven Linus Torvalds que inventa Linux por mero aburrimiento.
- Lean el libro de los dinosaurios (*Operating Systems Concepts*) de Abraham Silberchartz, con ese libro podrán entender absolutamente todos los conceptos del curso!!!

Gracias por su atención!



Mail: Yerko.ortizm@mail.udp.cl